Technical Report 1153

Design and Implementation of a Flexible Robot

Andrew Dean Christian

MIT Artificial Intelligence Laboratory

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AI-TR 1153	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Design and Implementation of a Flexible Robot		5. TYPE OF REPORT & PERIOD COVERED technical report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Andrew Dean Christian		8. CONTRACT OR GRANT NUMBER(s) N00014-86-K-0685
9. PERFORMING ORGANIZATION NAME AND ADDRESS Artificial Intelligence Laboratory 545 Technology Square Cambridge, MA 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Blvd. Arlington, VA 22209		12. REPORT DATE August 1989 13. NUMBER OF PAGES 163
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Office of Naval Research Information Systems Arlington, VA 22217		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

flexible
robot
vibration
control
design

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the design, implementation, and control of a robot that has low natural frequencies of vibration. Insights into the problems of designing joint and link flexibility are discussed. The deflection of the robot under gravity is correlated with the fundamental frequency of vibration. Different link geometries and materials are evaluated.

The robot has three rotary actuators and two links, with two actuators at the

Cont. on back

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Design and Implementation of a Flexible Robot

by

Andrew Dean Christian

Submitted to the Department of Mechanical Engineering on August 16, 1989 in partial fulfillment of the requirements for the degree of Master of Science in Mechanical Engineering.

Abstract

This report describes the design, implementation, and control of a robot that has low natural frequencies of vibration. Insights into the problems of designing joint and link flexibility are discussed. The deflection of the robot under gravity is correlated with the fundamental frequency of vibration. Different link geometries and materials are evaluated.

The robot has three rotary actuators and two links, with two actuators at the base and the third as an elbow joint between the links. The links are interchangeable and the joints have variable flexibility built in. The robot is controlled by three separate processors running on a VMEbus. A Sun 3/180 workstation provides the development environment. A collection of operator controls have been built including a panel to control the amplifiers and brakes, and a joystick for teleoperation. A PD servo loop runs the robot in either joint or cartesian space.

Results from experiments on the control of residual vibration are presented. Three different motions of the arm have been studied: a constant vibrational frequency move, a changing vibrational frequency move, and a cartesian move. Impulse prefiltering and slowly accelerating moves are compared and shown to be effective at reducing residual vibration.

Thesis Supervisor: Warren P. Seering
Professor of Mechanical Engineering

Acknowledgments

Many people contributed to the work described in this report, many more than I could list here. I'd like to thank my thesis advisor, Warren Seering, for his constructive criticism and enthusiasm that pushed this project through. It has been my pleasure to work with the students in our ever-changing research group. The Old Guard: Michael Caine, Steve Eppinger, Peter Meckl, Ken Pasch, Neil Singer, Karl Ulrich, Erik Vaaler, and Al Ward. The New Guard: Brian Avery, Lukas Ruecker, Kamala Sundaram, and Bruce Thompson. The Avant-Garde (UROP): John Columbo, Patrick Haluptzok, and Bill Singhose. Special thanks to Neil, who got me interested in flexible control and helped with the initial design, and Erik, who helped a great deal with the design and construction of the robot.

There are other people who really have to be mentioned here: Dave Siegel and Sundar Narasimhan for the Condor system and putting up with a lot of stupid questions. Dennis Phinney at Sun Microsystems who went above and beyond the call of duty to get me a working computer. The ladies who run the AI Lab: Priscilla Cobb, Cody Curtis, and Marilyn Melithoniotes who helped me survive the remarkable bureaucracy of MIT. Ron Wiken runs a great stock room and helped out with many small items. Peter Ning, Anita Flynn and Rod Brooks for the use of their Macintosh network while writing this thesis.

I would also like to generally state my thanks to the faculty, staff, and students of the MIT Artificial Intelligence Laboratory. It is a wonderful place to work.

Finally, there are three people who are particularly special. My parents have always provided support and sanity throughout the years. I owe them a great deal. And my good friend Divya, who has helped pull me through the crazy years of graduate study.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. This material is based upon work supported under a National Science Foundation Graduate Fellowship. Funding for the work was provided in part by the Office of Naval Research University Research Initiative Program under Office of Naval Research contract N00014–86–K–0685 and in part by the Charles Stark Draper Laboratory, grants #DL–H–285399 and #CSDL–5678.

Contents

List of Figures

List of Tables

Introduction and Literature Review

Chapter 1

This thesis deals with the design, construction, and testing of a flexible robot. We have built a three degree of freedom, two link anthropomorphic robot deliberately designed to have a lowest mode of vibration at 3 hertz. The purpose of this robot is to test control strategies that eliminate vibration. This chapter of the thesis discusses why we think this is a useful thing to do, summarizes what other researchers in the field have built, and explains the criteria we followed when designing the robot.

1.1 Why build a flexible robot?

When I explain to people what my research is about, the inevitable first question is "Why do you want to build a flexible robot?" Making a robot that vibrates doesn't make any sense. The stock answer to this question is that we didn't make the robot to be good, we deliberately made it to be as bad as possible. If we could control a truly awful robot, then we might be able to apply the same techniques to controlling systems that are not so bad. There are three categories of "not so bad" systems we are interested in.

First, many things vibrate in the real world. Controlling the vibrations of a robot or of machinery is not always a problem best solved by making the structure stiffer. There are many situations where it is not cost effective or even possible to stiffen up

the structure. For example: the head mechanism inside of hard disk drives, optical mirrors, fast automatic assembly equipment in factories, and heavy cranes. It might be possible to add stiffness to the system, but that adds cost and often slows down the speed of operation. A much nicer solution would be to use a smart control strategy that eliminated the vibration, but kept the motion quick.

Second, there is interest in space-based robots. NASA currently has several working robot arms used on the space shuttles; the Remote Manipulator System or RMS. An RMS is a fifty foot long, fifteen inch diameter robot that has an unloaded vibrational frequency of 0.5 Hz. When it carries a 15 ton satellite, this frequency drops by almost two orders of magnitude. Standard operating procedure is to command a move, allow the arm to reach the desired location, and then wait a few minutes for the vibrations to settle out before making the next move. Watching a robot arm vibrate is not a cost effective way to use an astronaut's time. Moreover, NASA is planning for the construction of a space station in orbit. Then the problem will not only be controlling the flexibility of the RMS arm, but also of the long beams that make up the structure of the space station. Not suprisingly, the aerospace community is actively working on ways to move objects around in space while keeping them from vibrating.

Third, there is the argument that by learning how to control systems that vibrate, we might be able to make robots lighter and faster. Many commercial robots are designed to be as stiff as possible in order to guarantee accurate endpoint control and a minimal amount of deflection from gravity. Being able to reduce the vibration in a system probably won't make the endpoint position more accurate, but it can cut down the weight of the links of the robot and allow an equivalent robot to be built with smaller motors and at a lower cost.

Those are our primary justifications for doing research in the area of flexible control. Our belief is that control strategies developed for use on our robot should map

directly over to space-based robots and might be adaptible to automatic machinery and similar problems.

1.2 Existing Flexible Test Systems

Interest in the control of flexible systems has blossomed in the last decade. Experimental rigs have been built all over the world. This section reviews some of the test equipment that has been constructed to study this question.

When you are trying to develop control algorithms for flexible systems a logical place to start is with a single flexible beam attached to a motor at one end and to a weight or payload at the other end. Literally dozens of researchers have set up this type of experiment, but perhaps the two oldest and best known are by Alberts [1] and Cannon [6]. This simple experimental setup still generates dozens of papers each year on the proper way to model the behavior of the arm and on the different control techniques that can be used. The primary advantages of the single beam system is that it can be modeled fairly easily with either classical techniques or finite elements, that it is simple to build, and by varying the length of the beam or the payload carried you can push around the vibrational frequencies. Normally the beam is made stiff in the direction of gravity and flexible perpendicular to gravity; this keeps the modes of vibration in the same direction as the motion of the motor which simplifies the control problem. A interesting and successful open-loop experiment using an arm that can vibrate in the direction of gravity as well as in the direction of control was done by Petterson [20].

The situation gets more complex when you begin experimenting with flexible structures that have more than one degree of freedom. One such setup is a single beam driven by a rotational joint and a prismatic joint, separately done by Yuh [30] and Koivo [13]. That is, the beam can both rotate in the plane and also be extended or shortened. Generally this is a thin, flexible beam attached to the end of

a Stanford/JPL arm. The primary advantage of this setup is the ability to change the frequency of vibration (by extending the arm) while keeping all actuators of the robot back at the base. A similar setup by Chalhoub [7] extends this to a three degree of freedom spherical coordinate manipulator. That is, a single flexible beam is attached to a base which has two rotational joints and one prismatic joint. The extra degree of freedom in this setup can be used to experiment with the effects of gravity, centripetal accelerations, and non-symmetrical links. To explain the centripetal and gravity forces, imagine a flexible beam with a mass at the end. Gravity acting on the mass pulls it down. When the robot turns, centripetal acceleration causes the mass to rise and starts the beam vibrating up and down as well as side to side. Controlling this vibration is potentially easier when you have a motor that can act in the same direction as the vibration.

When the flexibility issue is extended to two link manipulators, the experiments get more complicated. Consider a two link arm moving in a plane with two rotational joints, one at the base and one at the elbow. One of the first such arms used by Hollars [12] was designed so that the links themselves were stiff, but the joints had flexibility. This design has four good points: with encoders attached to the arm both before and after the flexibility in the joint the position of the end of the robot can be accurately calculated. The frequencies of vibration of the robot are dependent on the position of the elbow joint and can vary considerably throughout the workspace. The two revolute joints strongly resemble the structure of a traditional robot and allow the endpoint to be positioned arbitrarily in a plane. Finally, using joint flexibility is attractive when one considers that a great deal of the flexibility in modern robots comes from elasticity in the gears.

Another form of the two link manipulator is when you keep one of the links stiff and let the other one vibrate, such as in Oakley [19] and Schmitz [23]. This gives the more complicated behavior of distributed flexibility (and lack of knowledge of

the endpoint position) yet allows the researcher to use both joints to compensate for the vibrations. An interesting variant on the stiff/flexible link pair is to use one long flexible link with a fast (and stiff) end effector done by Tilley [27]. The two degree of freedom end effector was used to precisely control the position of the endpoint of the arm (with appropriate feedback) so the vibrations of the long flexible link do not affect what the arm is carrying.

The logical extension of the two link planar robot with joint or single link flexibility is to make a two link planar robot with both links flexible. Several research groups (including a number of the groups cited above) either have such an arm or are in the process of building one, two that I haven't yet mentioned are Bayo [2] and Chrétien [8]. The distributed flexibility, two link planar arm has the important characteristic of exhibiting several low frequencies of vibration that are dependent on the position of the arm. At the same time, by keeping the motion of the arm in a plane (and usually suspending the arm on air bearings) one does not have to deal with the arm sagging under the influence of gravity or vibrations out of the plane of motion. An additional advantage of the planar motion is that you can attach a bright light source to the arm and track the position of the arm using a camera and a frame grabber.

Finally, there are a few three degree of freedom robots designed to exhibit flexible behavior. Daniel [10] discusses three separate robots. The first is a traditional single beam on a motor. The second is a two link, two degree of freedom planar robot with two flexible links. The third is the Rotabot, a three link, three degree of freedom robot. Three revolute joints to drive the links. A stiff exoskeleton carries the bending loads, but rotates freely. The motors are connected to the joints by flexible shafts so the robot is stiff in bending but compliant in torsion. The advantage of using a stiff robot with flexible joints is that one gets the complicated motion of a three degree of freedom, non-planar robot with the ability to accurately measure where the robot

is in space.

The most complete implementation of a three degree of freedom flexible robot that I know of is by Pfeiffer and Gebler [21]. They have implemented a robot with three revolute joints and two links. The first two joints are together in the base and the third is the "elbow" of the robot. The joints are DC motors connected to harmonic drives with angular encoders to measure position. The links are rectangular in cross section to keep the arm from deflecting too much under gravity loading. For control feedback, they have added strain gauges along the arm to measure link deformation.

A final comment on flexible test fixtures: A great deal of work in flexible control is being done in the Aerospace field. Not suprising; they have to deal directly with large space structures made of thin components. Several research groups have been experimenting with models of the space shuttle holding a large antenna or truss. COFS (Control of Flexible Structures) (see [9,29]) is a 60 meter truss structure with actuators and a fancy tip package. They are planning on sending it up on the space shuttle to do experiments and take data on how it vibrates. The SCOLE program (Spacecraft Control Laboratory Experiment) (see Quinn [22]) is a precursor of the COFS project. A steel plate (representing the space shuttle) is hung by a cable and a ten foot tube with a mock antenna array hangs from the plate and simulates the effects of the COFS structure.

1.3 Design Criteria

The single flexible beam experiments and two flexible beams in a plane experiments are a fertile and important area of experimentation. They force you to deal with the complexities of controlling a flexible structure without getting too wrapped up in all of the complications that come with a three degree of freedom flexible robot. So what did we hope to gain by building such a structure?

The first and obvious answer is that the three degree of freedom structure more

closely mimics the RMS robot and other space robots that have yet to be created. If we can control our test fixture, I believe we have a good chance at controlling flexible space robots.

The second answer is that when you simplify the problem down to a single link or two in a plane, you lose some of the important control questions. Our robot exhibits extremely complicated modes of vibration (see Chapter 5). The vibrational frequencies shift by a factor of 1.7 as the robot pulls in or stretches out, and it is important to remember that the vibrations are not restricted to a plane. The lowest four modes of vibration are the easiest to see. Two of the modes are predominantly in the plane perpendicular to the axis of the elbow joint. The other two modes are predominantly in the direction of the axis of the elbow joint. Normal vibration comes when the motion of a joint induces vibration in the direction that the joint moves. But this is not the only way to induce vibration. When the base joint swings the arm around the centripetal acceleration starts vibrating the arm up and down. When the arm swings from a vertical position to a horizontal position, there is change in direction of the gravity force vector which affects the vibration of the arm. Most of the vibrational modes are directly controllable with a joint, but if the arm is vertical it has an uncontrollable mode of vibration in the direction that the elbow motor doesn't move. In general, the robot exhibits particularly complex forms of vibration.

The third answer for the question of "Why build something so difficult to control?" is because we are experimenting with some methods of control that apply to this arm. Chapter 5 shows some experimental data comparing input preshaping techniques developed by Neil Singer [24,25] with standard methods of moving robots.

Given that control of vibration is an interesting and worthwhile topic, we set out to build a test fixture that would challenge the best control theory. Thus, the MIT "Flexbot". The following is a summary of the design criteria for the Flexbot and some brief comments on why we think these criteria are relevant.

Anthropomorphic, 3 degree of freedom, 2 link robot: We choose this configuration because it demonstrates complicated vibrational behavior and because it is geometrically similar to many modern robots.

Lowest possible vibrational frequencies: We designed for a frequency of vibration around 2–3 Hz. Chapter 2 explains why this is a reasonable frequency range for a three dimensional arm and why lower frequencies are impractical in an earth-based robot.

Symmetry of vibrational modes and frequencies: To keep robots from deflecting under the influence of gravity, you can make the links of the robot stiffer in the direction of gravity. This allows low frequencies of vibration in motions perpendicular to gravity, but keeps the arm from sagging. It also limits the effects of vibration caused by centripetal acceleration or induced by the change in the gravity force vector as we move through the workspace. We deliberately choose to make the arm symmetric so that it has similar vibrational frequencies both in the direction perpendicular to gravity and in the direction of gravity.

Interchangeable links: A nice property of a flexible robot arm is that in order to get the low frequencies of vibration, the links are often just small metal bars. We designed the Flexbot so that unbolting one link and substituting another would be a simple affair.

Variable joint stiffness: In the RMS, approximately 50% of the flexure of the arm comes from flexibility in the gear box. We put springs into the joints of the Flexbot so that we could experiment with different amounts of flexibility in the joints.

Strength and speed: To properly excite multiple modes of vibration, we wanted a fast robot. So that we might experiment with payloads that change the

vibrational frequencies, we traded off some speed for torque. Our desired configuration had to be able to lift at least 15 pounds at full stretch and move from one end of the workspace to the other in 0.5 second. Chapter 3 covers in more detail the actual specs of the robot.

Size: Unfortunately, our laboratory does not have the space to hold a large robot. We had to limit the overall length of the robot to about 4 1/2 feet. Chapter 2 discusses some of the tradeoffs and problems inherent in a small vibrational robot.

Fast computation: To test complex control algorithms and watch the vibrational modes of the system we needed a fast computer system. We have a system that servos all 3 robot axes digitally at 1000 Hz and still has enough overhead left in it to play with fancy control algorithms. The computer system and software is discussed in detail in Chapter 4.

Modularity: We didn't want to be forced to always work with the robot running with all three axes. We designed the Flexbot so that it would be easy to use it with either one or two axes. In fact, our initial experiments were done with the robot running in a single axis mode and then with it running using just the two base degrees of freedom. It can also be configured to be a planar two degree of freedom, two link robot.

Backdrivability: For data recording and experimentation, we designed all of the joints of the robot to be highly backdrivable. This allows us to see the vibration of the end of the robot back at the joints and has possible future applications in force control.

Vibration Feedback: There are a number of popular ways that have been used to get feedback on the vibration of a flexible robot. The simplest is to measure

the deflection at the axis; this requires a backdrivable joint or a measuring device on the other side of the flexibility in the joint. Another way is to mount an accelerometer on the robot and feed back accelerations. Strain gauges along the length of a flexible beam can measure deformation. Some form of camera/optical feedback can measure the absolute position of the beam. In our case we opted to begin experimentation by using the information available from the joint positions (made practical by the Backdrivability criteria). In the future I suspect we will use accelerometers mounted along the robot as an additional source of information.

That completes the design criteria for the robot. The remainder of this thesis is broken up into the following chapters:

Chapter 2: Design suggestions for the issues involved in building a flexible robot. Gravity is a major problem when you try to build a robot that vibrates at low frequencies in three dimensions. This chapter also discusses material choice for the links and the design of flexibility at the joint.

Chapter 3: Hardware implementation of the Flexbot. A description of what the robot looks like, the power trains used to drive the joints, and what the links look like. We conclude by summarizing the performance specifications for the robot.

Chapter 4: The electrical side of the robot. A description of the computer hardware, operator interface to the robot, cabling, and computer software. The servo loop implemented to control the robot and the kinematic behavior of the arm.

Chapter 5: Experimental results. A summary of how the robot behaves in response to normal commands and some implementations of techniques we've tried that reduce the amount of vibration in the system.

Chapter 6: Conclusions and suggestions for future research that could be done using this arm.

Design Considerations

Chapter 2

2.1 Introduction

A flexible earth-based robot suffers from an inseparable problem: gravity. Unless you build in some form of compensation to eliminate the effects of gravity, your robot will sag. The standard way to get around sagging is to make the links of the robot asymmetric so that they have a higher stiffness vertically than horizontally. Unfortunately, directional stiffness can eliminate some of the very effects that you are studying. Consider a standard two link robot with three degrees of freedom: two revolute joints at the base and a revolute joint at the shoulder whose axis of rotation is parallel to one of the base joints. Rotation of the base excites vibration in both the radial direction and the direction perpendicular to this. Rotation of the upper two joints excites vibration primarily in the radial sense, with a negligible component perpendicular to this. What you have done by making asymmetric links is raise the stiffness (and therefore the natural frequencies) of your robot exactly in the direction that the majority of your joints move.

Our design criteria for the Flexbot specifies that the robot should have symmetrical links; that the vibrational frequencies in both the radial direction and the direction perpendicular to this should be approximately the same. This chapter deals with the underlying issues that came up while designing links for the robot

that could give the desired flexibility, resist the sagging influence of gravity, and not break. The first section deals with how the endpoint deflection of the robot under gravity relates to the lowest natural frequency of vibration of the robot. The second section talks about the material choice and shape of a flexible link. The third section presents a short discussion on joint flexibility.

2.2 Vibration Considerations

When building a robot that will have primary modes of vibration in all directions, a natural design issue is what will be the lowest natural frequency of the robot. A low fundamental frequency has several advantages and one clear disadvantage. The low fundamental frequency is easy to observe and record. It allows the higher modes of vibration to occur at frequencies that may also be visible. But the disadvantage is that a flexible arm will sag under gravity.

It turns out that the deflection of the arm under gravity is a very good way to estimate its natural frequency of vibration, and vice-versa. To demonstrate this, we will begin by deriving the relationship between the endpoint deflection of a single beam and its natural frequency. Then we demonstrate that for real two link flexible systems, the formula relating endpoint deflection to natural frequency forms a useful estimate of the system's natural frequency.

2.2.1 Single Beam Under Gravity Loading

There is a useful relation between the natural frequency of vibration and the deflection of a single beam under gravitational loading, mentioned by [5]. If we consider a single cantilever beam with a mass (as shown in Figure 2.1) and assume that it behaves as a Bernoulli-Euler beam we have

$$\delta_L = \frac{mgL^3}{3EI} \tag{2.1}$$

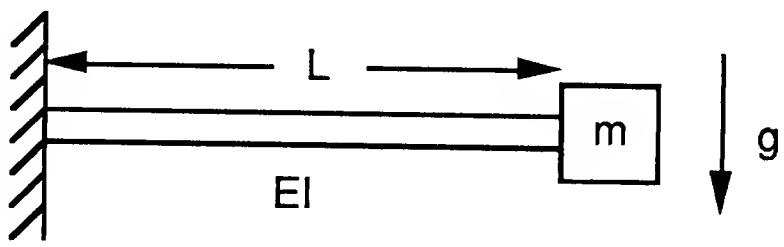


Figure 2.1: Single cantilever beam with a mass

where δ_L is the deflection of the end of the beam, E is the modulus of elasticity, I is the moment of inertia, and the mass of the beam is considered to be negligible. The spring constant of the beam that relates the endpoint deflection to the force acting at the end can be written as

$$K = \frac{mg}{\delta_L} \tag{2.2}$$

To a good approximation, the lowest natural frequency of the beam is given by

$$f_g = \frac{1}{2\pi}\sqrt{\frac{K}{m}} \tag{2.3}$$

where f_g is in Hertz. By combining Equations (2.2) and (2.3) we express the frequency of vibration of the cantilevered beam as

$$f_g = \frac{1}{2\pi}\sqrt{\frac{g}{\delta_L}} \tag{2.4}$$

Hence the natural frequency of vibration of the beam can be approximated as a function of its deflection under gravitational loading. We can also write the equation this way:

$$\delta_L = \frac{g}{4\pi^2 f_g^2} \tag{2.5}$$

or using $g = 9.8$ m/sec^2, we have

$$\delta_L = \frac{25}{f_g^2} \text{ centimeters} \tag{2.6}$$

$$f_g = \frac{5}{\sqrt{\delta_L}} \text{ Hz} \tag{2.7}$$

where δ_L is in centimeters and f_g is in Hertz. This is displayed graphically in Figure 2.2. Note particularly that frequencies under a few hertz result in extremely large deflections under gravity.

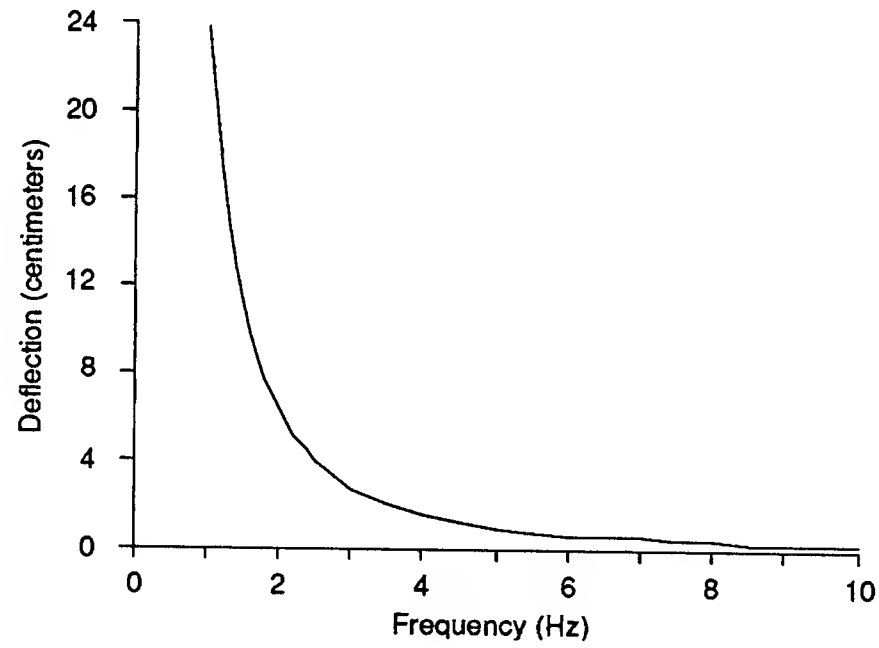


Figure 2.2: Deflection of a Single Beam under Gravity

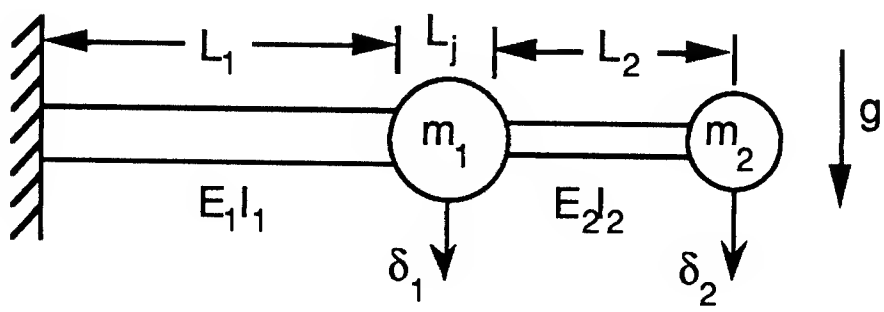


Figure 2.3: Two Cantilevered Beams with Masses

2.2.2 Two Beams Under Gravity Loading

Equations (2.6) and (2.7) are useful formulas to keep in mind when you are considering the behavior of a single link flexible robot. Now consider a two link robot modeled as a two beam system with a joint mass m_1 between the links and an additional payload mass m_2 suspended at the ends, as shown in Figure 2.3. To make this a realistic model of a robot arm, we include the length of the first mass as L_j. We simplify the analysis with two assumptions. We assume that the masses of the beams are negligible as compared to the masses of the joint and the payload. We assume that the beams behave as Bernoulli-Euler beams in bending. The flexibility

matrix of the system (see [26]) is

$$\begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \end{bmatrix} \tag{2.8}$$

where δ_1, δ_2 are the deflections at m_1 and m_2 from the forces F_1 and F_2, and a_{ij} refers to the deflection at mass i due to a unit force at mass j. The values of a_{ij} can be found as

$$a_{11} = \frac{1}{K_1}[1 + \frac{3L_j}{2L_1} + \frac{3{L_j}^2}{4{L_1}^2}] \tag{2.9}$$

$$a_{12} = a_{21} = \frac{1}{K_1}[1 + \frac{3L_j}{4L_1} + \frac{3}{2{L_1}^2}(L_1 + L_j)(L_2 + L_j)] \tag{2.10}$$

$$a_{22} = \frac{1}{K_2} + \frac{1}{K_1}[1 + \frac{3(L_2 + L_j)}{L_1} + \frac{3}{{L_1}^2}(L_2 + L_j)^2] \tag{2.11}$$

where we have substituted $K_1 = 3E_1I_1/{L_1}^3$ and $K_2 = 3E_2I_2/{L_2}^3$. Using the standard assumption of harmonic motion and replacing the forces F_1 and F_2 by inertia forces $F_i = -m_i\ddot{\delta_i} = \omega^2 m_i \delta_i$ we then find the vibrational frequencies by calculating the determinant and solving for ω from

$$\det \begin{vmatrix} (a_{11}m_1 - \frac{1}{\omega_n^2}) & a_{12}m_2 \\ a_{21}m_1 & (a_{22}m_2 - \frac{1}{\omega_n^2}) \end{vmatrix} = 0 \tag{2.12}$$

which can be solved explicitly for $f_n = \omega/2\pi$ as

$$f_n = \frac{\sqrt{2}}{2\pi}\Big(a_{11}m_1 + a_{22}m_2 + ((a_{11}m_1 + a_{22}m_2)^2 - 4(a_{11}a_{22}m_1m_2 - {a_{12}}^2m_1m_2))^{1/2}\Big)^{-1/2} \tag{2.13}$$

The endpoint deflection of this system due to gravitational loading is

$$\delta_{tip} = a_{21}m_1g + a_{22}m_2g \tag{2.14}$$

If we substitute this into (2.4), we get the approximation

$$f_g = \frac{1}{2\pi}\sqrt{\frac{1}{a_{21}m_1 + a_{22}m_2}} \tag{2.15}$$

This value can be compared to the expected vibrational frequency f_n from Equation (2.13).

In fact, it is easy prove that $f_g \leq f_n$. So, assuming that the frequency estimate f_n is fairly close to the actual frequency of the system, then the quick f_g calculation will be a lower bound for the actual lowest frequency of vibration of the system. As long as f_n is not too much greater than f_g, f_g forms a useful estimate of the natural frequency. In the next section, we will empirically demonstrate that f_g is a good estimate for real two beam, two mass systems.

Equations (2.13) and (2.15) depend fundamentally on the assumption that the system can be considered to be a long beam with a mass on the end. In fact, when $m_2 \to \infty$ or $K_1 \to \infty$, $f_g \simeq f_n$. So as the system more closely resembles either a single mass/beam system (where m_1 is completely negligible) or a system with just an end mass (where K_1 is so stiff that you can treat K_2 as a cantilever beam built into a wall), the closer it matches the ideal case. If you keep reasonably "balanced" values for your parameters, in the sense that each beam participates in the vibration and neither mass strongly exceeds the other, $f_n \simeq f_g$ is a good approximation.

2.2.3 Comparison of Vibrational Formulas

We would like to compare the values of f_g and f_n to determine the usefulness of the "tip deflection under gravity" approximation. The vibrational frequency f_n is a function $f_n = f_n(L_1, L_2, L_j, m_1, m_2, k_1, k_2)$ which is too complex to graph easily. Instead, we pick two sets of parameters and see how varying them affects the ratio of analytically calculated vibrational frequency to the gravitational estimate of vibrational frequency.

Two typical cases of beam configurations are displayed in Table 2.1, one aluminum and one steel. These values were chosen as representative of the types and sizes of systems that we have considered in the course of our research. Each of these

Link	L_1 (cm)	L_2 (cm)	L_j (cm)	m_1 (kg)	K_1 (N/cm)	m_2 (kg)	K_2 (N/cm)
Steel	45	45	20	7	220	1.4	12
Aluminum	60	45	10	4.5	210	4.5	80

Table 2.1: Parameter Values for Beam Comparisons

robot configurations has a natural frequency of vibration of approximately 3.5 Hertz. Additionally, the K values were choosen so that each beam participates equally in the vibration; that is, if you assume a round cross section of beam, the maximum stress level experienced in beam L_1 under gravity loading is the same as that experienced in L_2.

Figures 2.4 and 2.5 compare the gravity estimate of the natural frequency to the analytical estimate of the natural frequency of a two link arm. The gravity estimate is always lower, so the graphs are plotted as the percentage the analytical estimate (or "Actual" frequency guess) is of the gravity estimate. A rating of 100% would mean that the gravity estimate was exactly right. As shown in Figures 2.4 and 2.5, varying the lengths of the first and second links does not change the error of the gravity estimate by more than a few percent. The nominal error for the steel links is about 25% and for the aluminum links is about 9%. The aluminum link frequency estimate is better because of the large second mass relative to the first mass. If we then hold the lengths constant and change the weight of the first mass and the stiffness of the first link, as shown in Figures 2.6 and 2.7, the error in the estimate varies more than from changing the link lengths, but the error still stays to within 10% of the original 25% higher for the steel links and 10% higher for the aluminum. Changing the first mass has the largest effect on the estimate but even then f_g is an estimate good to within 20% as long as m_1 is kept small.

After trying different values of lengths, masses, and stiffnesses, it becomes evident that the estimate of vibrational frequency based on static gravitational deflection is

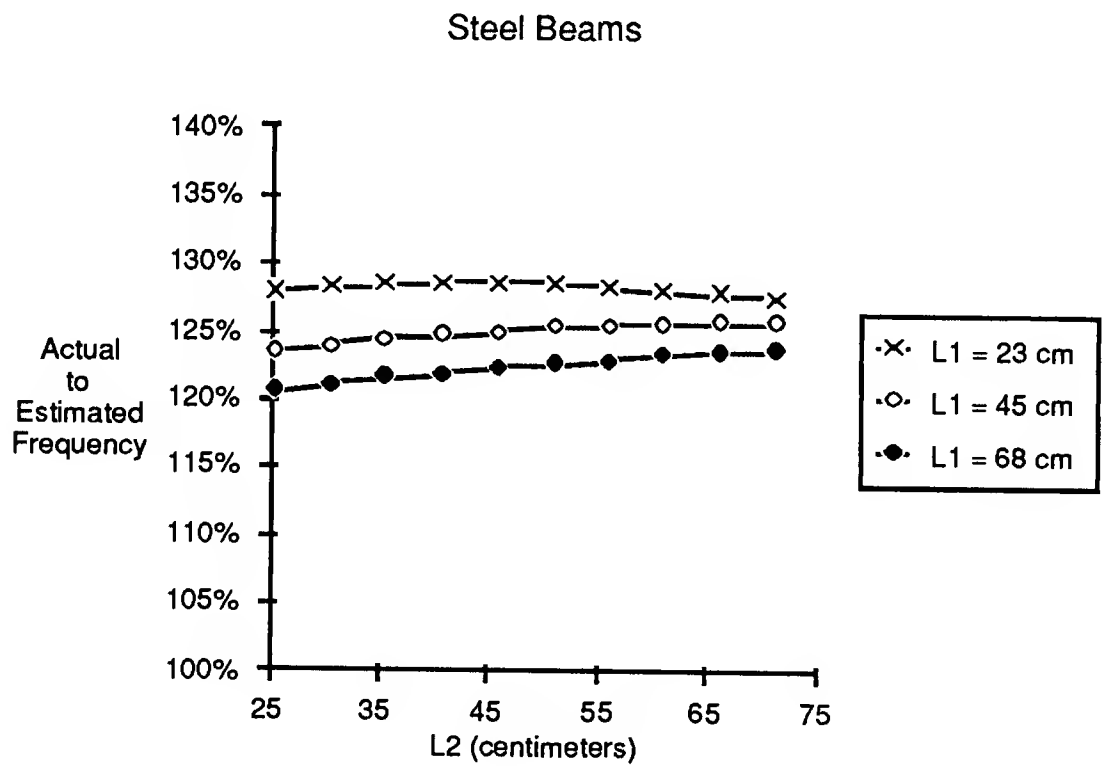


Figure 2.4: Variation in the ratio of frequency estimate to actual natural frequency as a function of link length

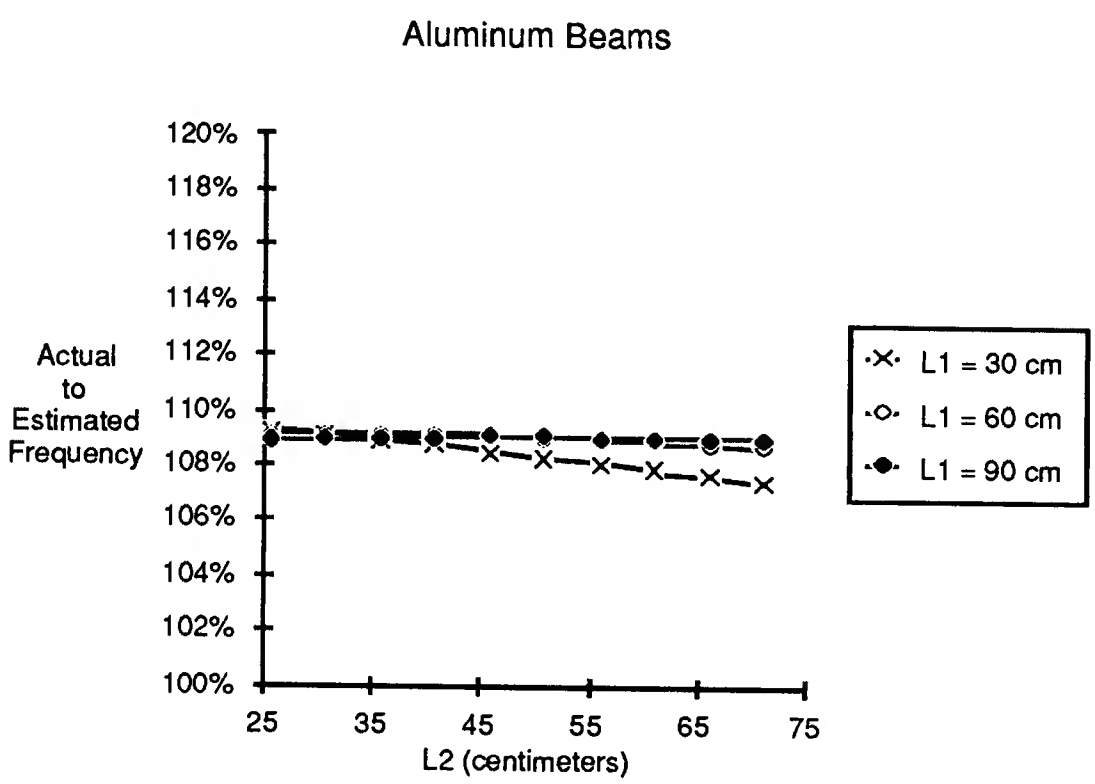


Figure 2.5: Variation in the ratio of frequency estimate to actual natural frequency as a function of link length

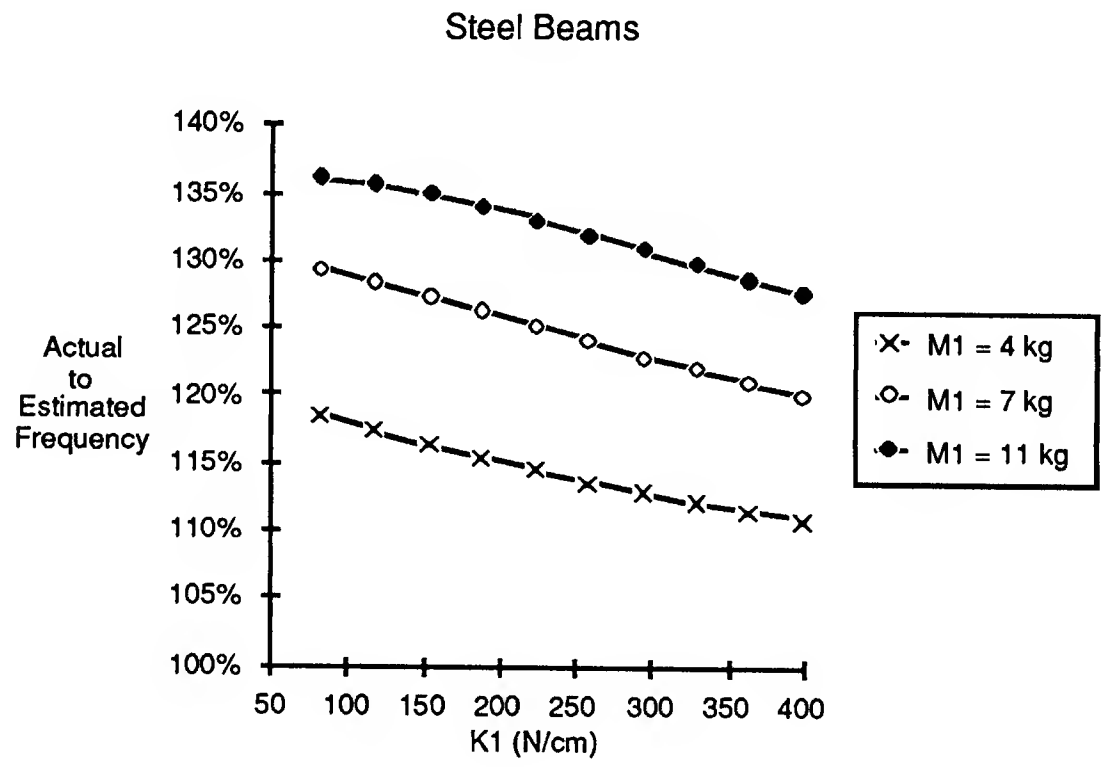


Figure 2.6: Variation in the ratio of frequency estimate to actual natural frequency as a function of first link stiffness and joint mass

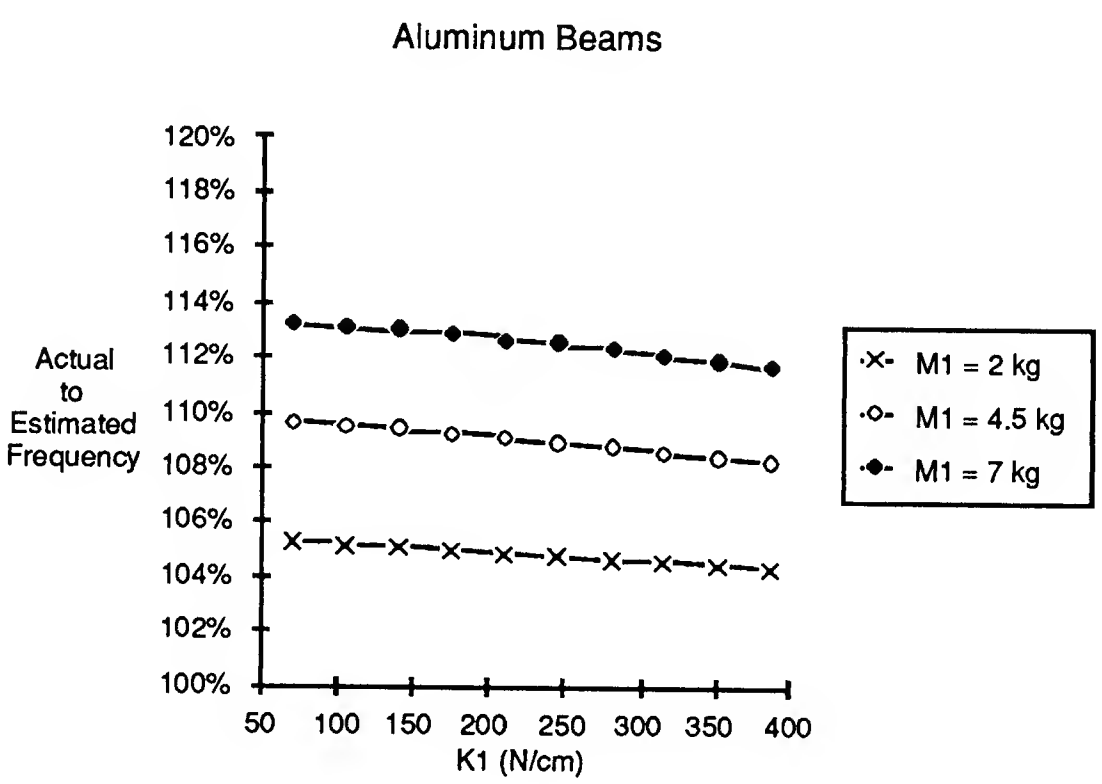


Figure 2.7: Variation in the ratio of frequency estimate to actual natural frequency as a function of first link stiffness and joint mass

very good. To a reasonable approximation, it is safe to say that the vibrational frequency estimate based on endpoint deflection under gravity loading of a typical robot is good to about 30% of the actual value. As f_g forms a lower bound to the vibrational frequency, if you want a system with the lowest vibrational frequency for a given endpoint deflection, you can't do any better than $5/\sqrt{\delta_L}$ (see Equation 2.7). This result is useful because it is often easier to calculate an endpoint deflection for a complex system than a resonant frequency.

It is important to realize that the gravitational deflection versus natural frequency relation does not depend on the length of the beams (see Equation 2.6). If you want a system that vibrates at 1 Hertz, you must live with a endpoint deflection of at least 25 centimeters under gravity, regardless of how long your beams are. Fortunately, the required endpoint deflection falls off quickly as the vibrational frequency is increased (see Figure 2.2). The robot we built had a target frequency of 3 hertz, which gives it a little over a one inch endpoint deflection due to gravity.

Finally, remember that this relationship has been derived based on the endpoint deflection under a gravitational load. Estimates of the vibrational frequency in other directions are found just as easily if you assume that the robot has been placed on its side and calculate how far the endpoint deflects.

2.2.4 Length of the Arms

If you are designing a robot that will have its fundamental frequency of vibration at one hertz, the robot is going to sag under its own weight at by least 25 cm. A one meter arm is too short to get any kind of linear behavior. But if you want to have the end of the robot vibrating with an amplitude of 5 centimeters, how flexible should you make the links? We can calculate the level of stress inside of a link for a given endpoint deflection.

In terms of a single mass system with a constant cross-section beam and a given

tip deflection δ_{tip}, we write the equation

$$\sigma = \frac{My}{I} \tag{2.16}$$

where M is the bending moment in the bar, y is the maximum distance from the neutral axis and I is the moment of inertia. The bending moment is

$$M = FL = K\delta_{tip}L = \frac{3EI}{L^2}\delta_{tip} \tag{2.17}$$

where K is the stiffness defined in Equation 2.2, F is the force required to produce an end deflection δ_{tip} and L is the length of the beam. Combining Equations (2.16) and (2.17), we have

$$\sigma_{max} = \frac{3Ey}{L^2}\delta_{tip} \tag{2.18}$$

Equation (2.18) shows that the stress level in the link is directly proportional to the tip deflection and inversely proportional to the square of the length. To keep the bending stresses to a minimum for a given tip deflection and natural frequency, you should make the link as long as possible.

2.3 Link Types

The first step in designing a flexible robot normally includes deciding how long the links will be, what the lowest natural frequency of vibration will be, how much torque will be available at the joints, and what the approximate weights and sizes of the joints will be. Once these numbers have been selected, you need to pick the actual shape of the links and material used in the links. There is a tradeoff here between the three key parameters: stiffness, mass, and stress level. Typically the stiffness will have been set by the decision of what the lowest frequency of vibration will be. Then you must design links for this stiffness while keeping the stress level and mass of the link to a minimum.

2.3.1 Springs

For our robot, in the initial design it worked out that a 3 hertz vibration mode put uncomfortably high stresses on the links. At the time we were doing our calculations based on links made out of solid bars of aluminum. One option that we looked at to eliminate this undesirable stress level was to replace the aluminum bar with a coil spring. At first glance the coil spring seemed to be the perfect flexible link; not very stiff and able to undergo large deformations without yielding.

It turns out that springs are *too* flexible for their weight. We can compare the weight and flexibility of a straight link made out of a bar of metal with diameter d to a spring made out of the same bar with wire diameter d, but coiled into a helix. Consider the overall lengths of the two links to be the same.

First, we compare the bending stiffness of the spring to the bending stiffness of the bar. The general equation of bending for a spring is

$$\kappa = \frac{\tau}{\theta} = \frac{Ed^4p}{32LD}\left(\frac{1}{1+E/2G}\right) \tag{2.19}$$

where τ is the bending moment, θ is the angle of deformation, p is the pitch of the spring, L is the length, D is the pitch diameter of the spring and E and G are the modulii of elasticity and rigidity. If we assume small angles of deformation, so that $\theta = dy/dx$ and we assume that the spring is fixed at one end and is loaded at the other end by a force F perpendicular to the length of the spring, then we write the deflection of the end of the spring as δ, where

$$\delta = \int_0^L \frac{\tau}{\kappa}\,dx = \frac{F}{\kappa}\int_0^L (L-x)\,dx \tag{2.20}$$

or

$$\delta = \frac{FL^2}{2\kappa} \tag{2.21}$$

Note that δ is not the axial deflection of the spring but a measure of how far the spring bends.

Now we find the effective spring constant of the spring as

$$K_s = \frac{F}{\delta} = \frac{Ed^4p}{32L^3D} \tag{2.22}$$

where we have made the approximation $E = 2G$. For the round bar with the same diameter d, we write

$$K_{bar} = \frac{3EI}{L^3} = \frac{3\pi Ed^4}{64L^3} \tag{2.23}$$

Combinining Equations (2.22) and (2.23), we find that

$$\frac{K_s}{K_{bar}} = \frac{2p}{3\pi D} \tag{2.24}$$

For a practical spring, it is safe to assume that $p < D$, so K_s is at most 20% of K_{bar}.

The weight of the spring can be compared to the weight of the straight bar . The weight of the spring is given by

$$W_s = \rho\pi d^2 L\sqrt{1 + \left(\frac{\pi D}{p}\right)^2} \tag{2.25}$$

and the weight of the straight bar is

$$W_{bar} = \rho\pi d^2 L \tag{2.26}$$

Then, if we assume that $(\pi D/p)^2 >> 1$, we get

$$\frac{W_s}{W_{bar}} = \frac{\pi D}{p} \tag{2.27}$$

Again, it is safe to assume that $p < D$, so the weight of a spring made of a coil of wire is at least 3 times the weight of a link made of a straight piece of that wire.

We conclude that the for a given cross-section d, the spring weighs at least three times as much as the straight bar and has at best 20% of the bending stiffness. If you want a very low bending stiffness and don't care about weight, the spring is the way to go. But if you want a given stiffness, then the spring is going to be at least 15 times as heavy as a straight bar of metal. Even though this allows you to not worry about breaking your link, the additional weight penalty in most systems is prohibitive.

2.3.2 Material Choice

For a given bending stiffness, the material used for the link will determine its size, weight, and how much deflection it will undergo before yielding. A material with a high yield stress is not necessarily the best choice. If it has a high modulus of elasticity, the link will have to be thinner to get the same bending stiffness. The smaller moment of inertia will result in a higher stress level than that of a link made of a material with a high yield strength and a low modulus of elasticity. A similar situation exists with weight; materials that have lower modulus of elasticity tend to have lower density than materials with a high modulus of elasticity. But the low modulus of elasticity requires a thicker link to get the same bending stiffness, so the weight of the link goes up. Hence using a material that is denser does not necessarily give a heavier link.

To compare materials, we first assume that the link is a solid bar of metal with a round cross section of diameter d. We choose a round cross section for two reasons: First, it has the same stiffness when bent in any direction, where the stiffness is the deflection of the end of the link with respect to a force applied at the end. Second, for a given bending moment and stiffness K, a solid bar experiences a lower bending stress than a hollow bar. (Consider two bars with the same moment of inertia; one solid and one hollow. The hollow bar will have a larger diameter and the bending stress is directly proportional to the diameter.) If we assume that we know the length of the link, the stiffness, and the applied bending moment, we can calculate the stress and weight of the link as a function of density, modulus of elasticity and yield strength.

Start with the standard formulas

$$\sigma = \frac{Md}{2I} \tag{2.28}$$

and

$$K = \frac{3EI}{L^3} \tag{2.29}$$

where E is the modulus of elasticity of the metal, M is the applied bending moment, I is the moment of inertia of a round cross section and L is the length of the beam. Rewrite Equation (2.29) as

$$I = \frac{L^3K}{3E} = \frac{\pi}{64}d^4 \tag{2.30}$$

or

$$d = \sqrt[4]{\frac{64L^3K}{3\pi E}} \tag{2.31}$$

To find the stress in the beam, combine Equations (2.28), (2.30) and (2.31)

$$\sigma = \frac{6M}{KL^2}\sqrt[4]{\frac{KE^3}{12\pi L}} \tag{2.32}$$

Equation (2.32) shows that with L, K and M fixed for a beam, the material with the lowest fraction of stress to yield stress will be the one with the largest value of ψ, where ψ is defined as:

$$\psi = \frac{\sigma_{yield}}{\sqrt[4]{E^3}} \tag{2.33}$$

The larger the value of ψ, the larger the bending moment the link will withstand before yielding.

Another material comparison is the weight of the bar as a function of the length and the spring constant. Using Equation (2.31) and taking ρ as the density of the link, we get

$$W = \frac{\pi\rho d^2 L}{4} = 4\rho\sqrt{\frac{\pi L^5 K}{3E}} \tag{2.34}$$

where W is the weight of the link. For a given L and K, the beam with the lowest weight will be the one that minimizes ϕ, where ϕ is given by:

$$\phi = \frac{\rho}{\sqrt{E}} \tag{2.35}$$

The larger the value of ϕ, the heavier link.

Material	σ_y (MPa)	E (GPa)	ρ (kg/m^3)	ψ	ϕ
1100–0 Aluminum	34	72	2800	1.4	330
2014–T6 Aluminum	415	72	2800	16.8	330
6061–T6 Aluminum	275	72	2800	11.1	330
1015 Steel	324	207	7700	5.9	535
4140 Steel	655	207	7700	12.0	535
Magnesium Alloy	240	45	1800	13.8	268
Titanium Alloy	830	114	4400	23.4	412

Table 2.2: Link Material Comparison

The optimal link material would have a large ψ and a small ϕ. In practice, no material is optimal although Titanium comes close. Table 2.2 shows the value of ψ and ϕ worked out for a number of different common metals. Titanium has the best strength to stiffness ratio and 2014–T6 Aluminum is a practical, lightweight alternative although its resistance to fatigue is limited.

2.4 Joint Flexibility

Robots have two types of flexibility; in the joints and in the links. Joint flexibility appears to the system as springs in series with the links. To illustrate this, look at the two link model shown in Figure 2.3. Now add in a torsional spring of value κ_1 where the first link meets the wall and a torsional spring of value κ_2 between the links. Equations (2.9), (2.10), and (2.11) relate the deflection of a point of the robot to a unit force applied at a one of the points, so we can modify them to take into account the torsional springs:

$$a'_{11} = a_{11} + \frac{L_1 + L_j/2}{\kappa_1} \tag{2.36}$$

$$a'_{12} = a'_{21} = a_{21} + \frac{L_1 + L_j/2}{\kappa_1} \tag{2.37}$$

$$a'_{22} = a_{22} + \frac{L_1 + L_2 + L_j}{\kappa_1} + \frac{L_2 + L_j/2}{\kappa_2} \tag{2.38}$$

These flexibility values can be substituted in Equations (2.13) and (2.15) to give the estimates for the lowest natural frequency of the system. We still have f_g as a lower bound for f_n, and assuming that the springs are not extremely flexible, f_g still gives a good approximation of the lowest natural frequency.

2.5 Conclusion

The endpoint deflection of a robot in a gravity field due to bending in the links and flexiblity in the joints provides an estimate of the natural frequency of vibration of the system. When all of the links are flexible and the weight of the payload is not negligible in comparison to the weight of the joints, this estimate is very close to the true frequency. As this estimate forms a lower bound, if your goal is to minimize the fundamental natural frequency of the system, you should make links that are as long as possible and have small masses at the joints. The "tip deflection under gravity" calculation is not restricted to the vertical case; we can put an imaginary gravitational field pointing in any direction, calculate the deflection of the robot, and estimate the natural frequency of vibration.

For maximum flexibility with minimum stiffness, the best material choice for the links is titanium. Some aluminum alloys may be inexpensive alternatives, but the fatigue characteristics of aluminum are unacceptable in most applications. Coiled springs give very little stiffness in comparison to their strength, but their weight makes them unusable in all but planar applications.

There are physical difficulties to building a test fixture with a very low frequency of vibration. To keep stress levels low in a flexible robot, long links are prefered, but may not be possible if the robot must fit inside of a laboratory. The longer links have a larger moment of inertia and need larger motors to drive them around.

Joints that contain motors carry a great deal of weight in the form of motors, gears, brakes, encoders, and ball bearings. Trying to optimize the robot to have low joint masses, long links, and adequate torque at the joints is an interesting task. With these constraints in mind, Chapter 3 details the hardware implementation of the flexible robot.

Hardware Design

Chapter 3

The robot design we settled on was a three degree of freedom robot with two revolute joints at the base and a revolute joint at the elbow. This chapter deals with the hardware design and implementation of the robot. It covers the power trains for each of the three joints, the implementation of the flexible links and the flexibility in the joints. The conclusion, Section 3.5, gives a summary of the performance of the robot.

3.1 General Specifications

Our target frequency for the lowest mode of vibration of the arm (using the flexible links alone) was 3 hertz. The remarks in Section 2.2.4 explain why making the arm as long as possible is desirable—for a given vibrational frequency, a short arm is subjected to a higher bending stress than a long arm. The limit to the size of the robot became the space available in our laboratory. To make it fit, we constrained the length of the robot to 52 inches.

Setting the length of the arm and taking into account the approximate size of the elbow joint gave us a working number for how long the flexible portion of each link could be; approximately 19 inches. Spreadsheet calculations determined that a 3 pound payload was optimal. A larger payload put too much stress on the links, and a smaller payload would not have given us the ability to radically alter the frequency

characteristics by removing the payload. Given the size of the payload and our desire to be able to accelerate the end of the arm at 5–6g's (g = acceleration due to gravity), the specifications for the elbow joint and the base followed naturally.

A picture of the robot appears in Figure 3.1. Two of the three joints are visible, the third is contained within the base the robot stands on. A side view of the robot base can be seen in Figure 3.2. The base has three purposes; it makes it easier for a human to work on the robot, it holds the isolation transformers and amplifiers for the motors, and it increases the workspace of the robot by allowing the robot to reach down below the height of the table surface. An additional advantage of designing the base to have a hidden, contained axis of rotation is that we can unbolt the upper two joints of the robot and use the base for simple experiments with a single flexible beam.

From Chapter 1 you might recall that we desired flexibility in the joints as well as in the links. We initially tried coiled springs, flexible rubber couplings, and cantilevered beams. None of these were satisfactory. We finally hit upon the idea of using stacks of Belleville spring washers. They have several advantages. They are small in size and very stiff. The cumulative stiffness of a stack can be easily varied by adding or removing washers or by changing the stacking configuration. They are cheap and available in a number of different metal thicknesses. Each of the three joints of the robot has a spring coupling of this form. Later sections will discuss the implementation of the flexible couplings.

A final comment on the general design of the robot: specifically, nomenclature. The robot has three axes, two at the base and one at the elbow. I refer to the joints as the *base axis*, *first axis* and *second axis or elbow joint.* This can be confusing to people. Why didn't I call them joints 1, 2 and 3? There are two reasons, one historical and one practical. The historical reason is that when we first began using this robot in November of 1988 we only had the axis within the base running. At

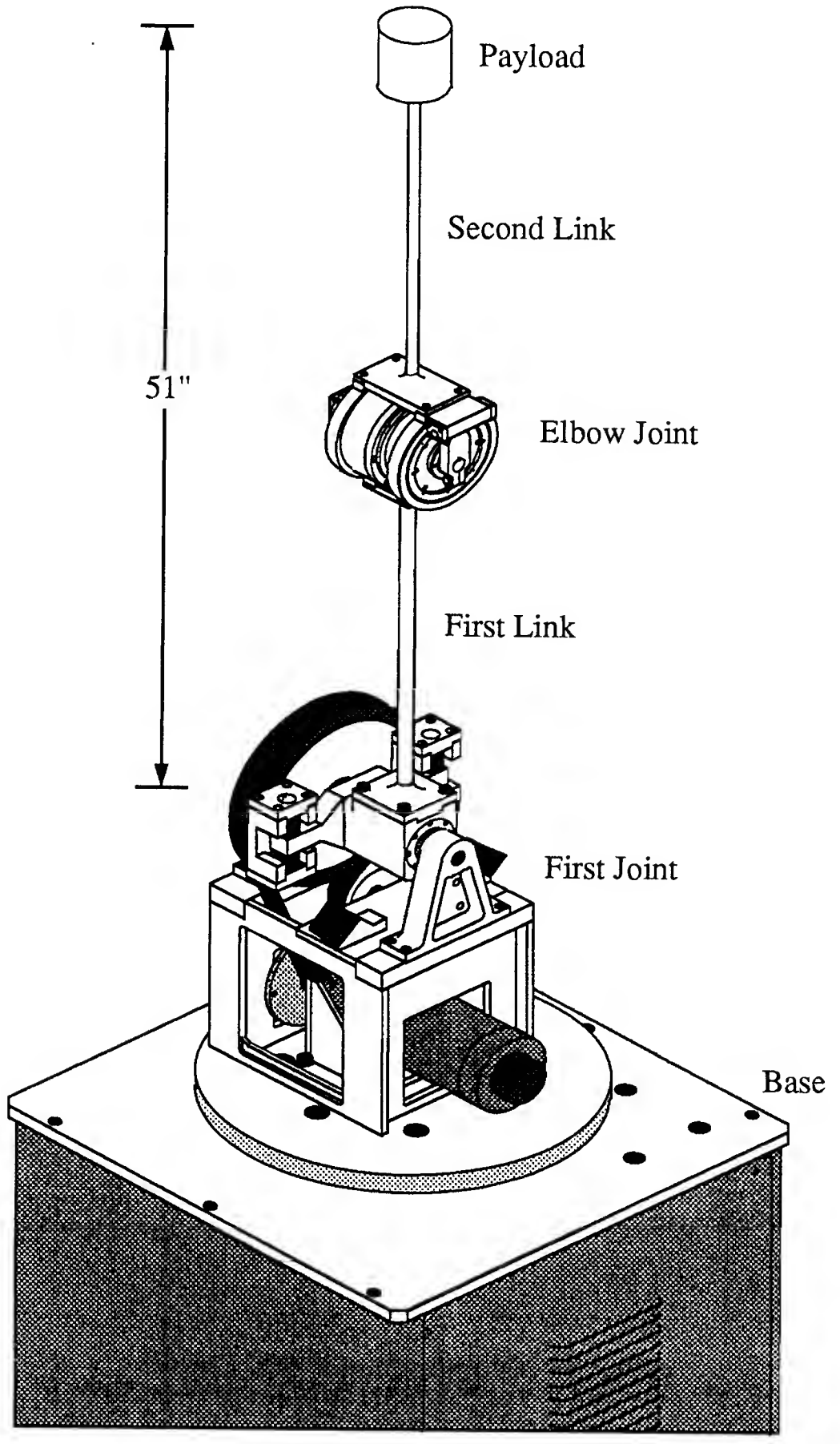


Figure 3.1: Isometric of the Flexbot. This view shows the robot outfitted with the flexible steel links and the nominal 3 pound payload.

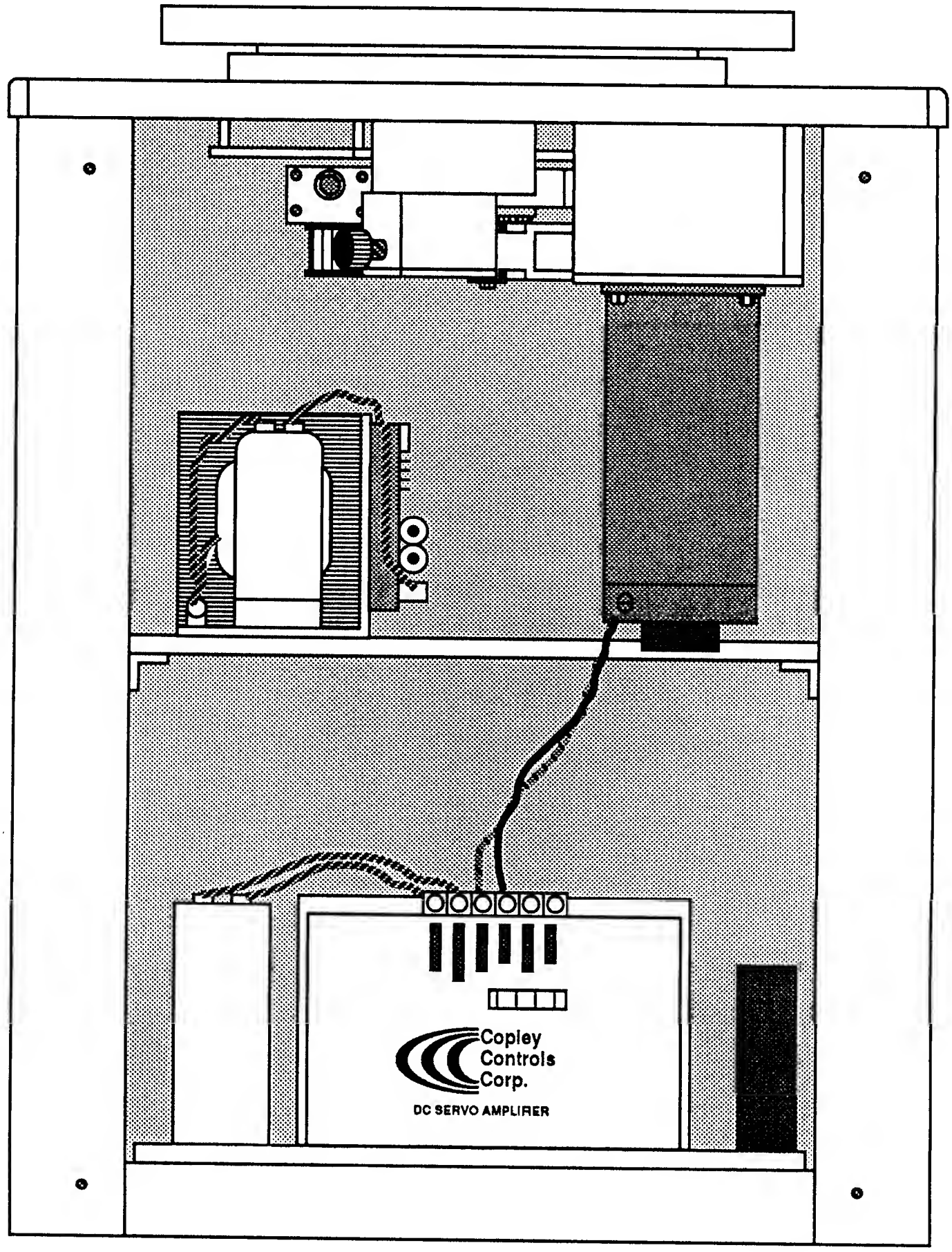


Figure 3.2: Side view of the robot base with the access door removed.

the time we naturally called it the "robot base". The practical reason is that the computer refers to the joints as 0, 1, and 2. Being a programmer by nature, I find it more natural to think of the joints as being labeled from 0 to 2 rather than do mental conversions every time I'm working on the code. However, "zeroth joint" doesn't roll off of the tongue easily, so I use the term *base axis.*

3.2 Base Design

This section details the design of the two joints of the robot located at the base. Section 3.3 will discuss the design of the elbow joint. I discuss these two joints at the same time because in many respects they are identical. They use the same motors, gear reductions, and other basic hardware.

3.2.1 Power Train

The basic design of the base axis can be seen in Figure 3.3. Rather than purchase an expensive direct drive motor, we opted for a large DC servo motor connected to the output by a 10:1 timing belt reduction. This design gives good acceleration, a high top speed and good backdrivability. The timing belt minimizes the backlash of the system. The timing belt does have a small amount of flexibility associated with it, but this is not important because it is driving a flexible coupling that contains a great deal of flexibility. Finally, a fail-safe brake is attached to each motor. Without the brakes gravity would cause the robot to collapse when power was removed.

Each of the two base axes uses an Indiana General 4050D-26 permanent magnet DC servo motor. The Indiana General has a continuous rating of 90 volts, 11.2 amps and 2400 RPM. It has a torque constant of 77.4 oz-in/amp or .403 ft-lb/amp and a voltage constant of 16.6 rpm/volt or 1.74 (rad/s)/volt. With our amplifier, we can supply it with up to 60 amps at 100 volts, giving a peak rating of 24 ft-lbs or a

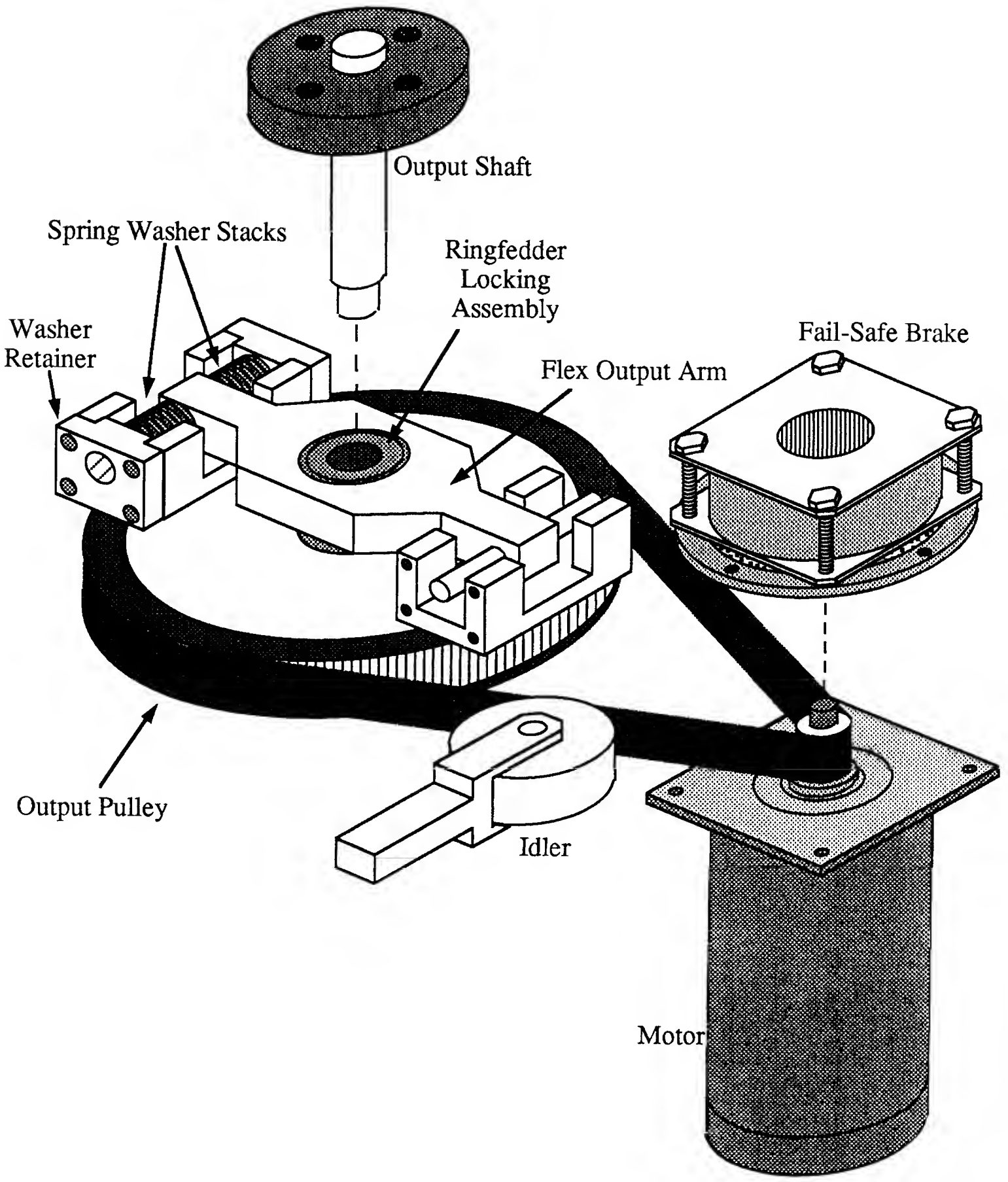


Figure 3.3: Isometric of the base joint drive train. Not shown is the motor and idler mounting hardware.

continous rating at 11.2 amps of 4.5 ft-lb, and a top speed of 2600 rpm. The motor weighs 13.75 pounds.

To power the motors we use a pair of Copley Controls Corporation Model 241 Pulse Width Modulated switching amplifiers. The 241 switches at 22 kHz and takes an input voltage of 24 to 165 VDC. We have it mated with a rectifier/capacitor circuit (also made by Copley Controls) that supplies 100 volts of DC current. The rated bandwidth of the amplifier is -3 dB @ 1000 Hz and the slew rate is 380 amp/ms. A model MB6 test board attached to the amplifier provides adjustable current limits (3–60 amps), gain, and balance. The 241 is rated at 60 amps for 2 seconds and 30 amps for continous operation. It is controlled by a differential ± 10 volt signal and has a inhibit line which only allows the amplifier to run when it is pulled low. The 241 has a reliable over-current limit with a reset switch and indicator lights to indicate amplifier status. It also provides a current monitor with feedback at 0.1 volt/amp.

Each motor has an HEDS HP–6000 optical encoder package attached to the back and an Electroid Fail-Safe MFSB–42 brake attached to the pinion shaft. The codewheel has 1000 counts per revolution, so with quadrature and the 10:1 gear reduction, the encoder can resolve at 40,000 counts per revolution of the output or to within 0.009 degrees. The fail-safe brake consists of a cork pad riding between two spring loaded steel plates. The plates are held apart by an electromagnet, so when power is removed, the brake closes. Each brake requires 0.44 amps at 24 volts, has an effective holding torque of 167 ft-lbs at the output of the gear reduction and weighs 4.0 pounds. An unexpected benefit of using this type of brake was the gentleness of the action. In an emergency stop, the brake closes but the cork pad can still slip. This limits the deceleration felt by the robot and prevents it from damaging the flexible components.

The motors drive a 10:1 Gates HTD timing belt reduction, the biggest reduction

we could conveniently fit on the robot. We ran a spreadsheet simulation with a reasonable guess of the inertia of the robot at full stretch and calculated the time required for the motor to complete a 30 degree move and a 180 degree move. The optimal reduction for the 30 degree move was a 24:1 reduction (0.139 seconds) and the optimal for the 180 degree move was a 14:1 reduction (0.429 seconds). Both of these are unreasonable to do in a single stage. Our backdrivability requirement made more than one stage out of the question. We ended up taking the smallest pinion we could fit on the motor shaft and the largest pulley that seemed reasonable—giving a 10:1 reduction. And the 10:1 is quite good, having a 30 degree move time of 0.188 seconds and a 180 degree move time of 0.461 seconds.

To maximize the gear reduction for the given space we used a 5mm pitch belt with a custom 15 tooth pinion that barely fits on the output shaft of the motor. A standard 25mm wide belt drives the 150 tooth output pulley. This design is outside of the rated specifications of the timing belt. However, the specifications for the Gates timing belts are based on infinite life, continuous operation at the rated torque, and generous safety margins. We talked to an engineer at Gates who agreed that the absolute strength of the belt is on the order of a few thousand pounds. To test this we purchased a belt and broke it with a hydraulic press. The experimental tensile strength of the belt was approximately 3000 pounds. Our motor at maximum torque can put about 600 pounds on the belt, so even with belt pretension we are under the yield strength. We do not expect fatigue problems as this robot is a test fixture and probably won't run for more than a few hundred hours. As a final test, I had the motor toss around a large inertia and then slammed it into the base (actually, this wasn't an *intentional* test). The belt failed, but the failure mode was the shearing off of two teeth. It did not fail by breaking, so it was still holding the robot up. The conclusion is that given the intermittent nature of normal operation the belts should be sufficiently strong. Moreover, the failure mode is safe and doesn't result in the

loss of control of the robot.

The timing belts are tensioned by a simple idler, best seen in Figure 3.3. The idler is a 3 inch diameter aluminum roller riding on a Torrington needle bearing. It is held in place by a "tuning fork" which rides in a square slot milled in a block of aluminum. Tension in the belt is supplied by a 1/2" bolt that thrusts the idler forward. A cover plate holds the idler in place in the block and can be tightened down to relieve some of the load from the 1/2" bolt.

The large output pulleys spin freely on 5204 double-row bearings. They connect to the outputs of the joints via the flex output arms. The couplings consist of four stacks of Belleville spring washers. The spring washers are pre-compressed by steel washer retainers and ride on dowel pins. The output arm (with the dowel pins) connects to the output of the axis. The designs of the output couplings and remainder of the joints is different for the base axis and the first axis, so they will be treated separately in the next two sections.

3.2.2 Base Joint Design

The base joint of the robot is pictured in Figures 3.2, 3.3, and 3.4. In this joint, the output arm of the Belleville spring washers is fixed onto a 1 inch steel shaft. The pulley rides on a bearing on the same shaft. The output arm attaches to the shaft via a Ringfedder RfN 7013.1-IN locking assembly. The Ringfedder is a single-taper self-aligning coupler that can transmit 323 ft-lb of torque. The shaft is silver-soldered (that is, brazed) into a 4 inch steel disk which is bolted to the output platter.

The output platter is an 11 inch diameter, 1 inch thick aluminum disk riding on the inside of an 11 inch Kaydon JU Type X sealed bearing. The outer ring of the Kaydon rests on the surface of the table and is held in place by a retaining ring (see Figure 3.4). The Kaydon has a dynamic rating of two tons axially and 750 ft-lb of torque, sufficient for our application. The output platter rides slightly higher than

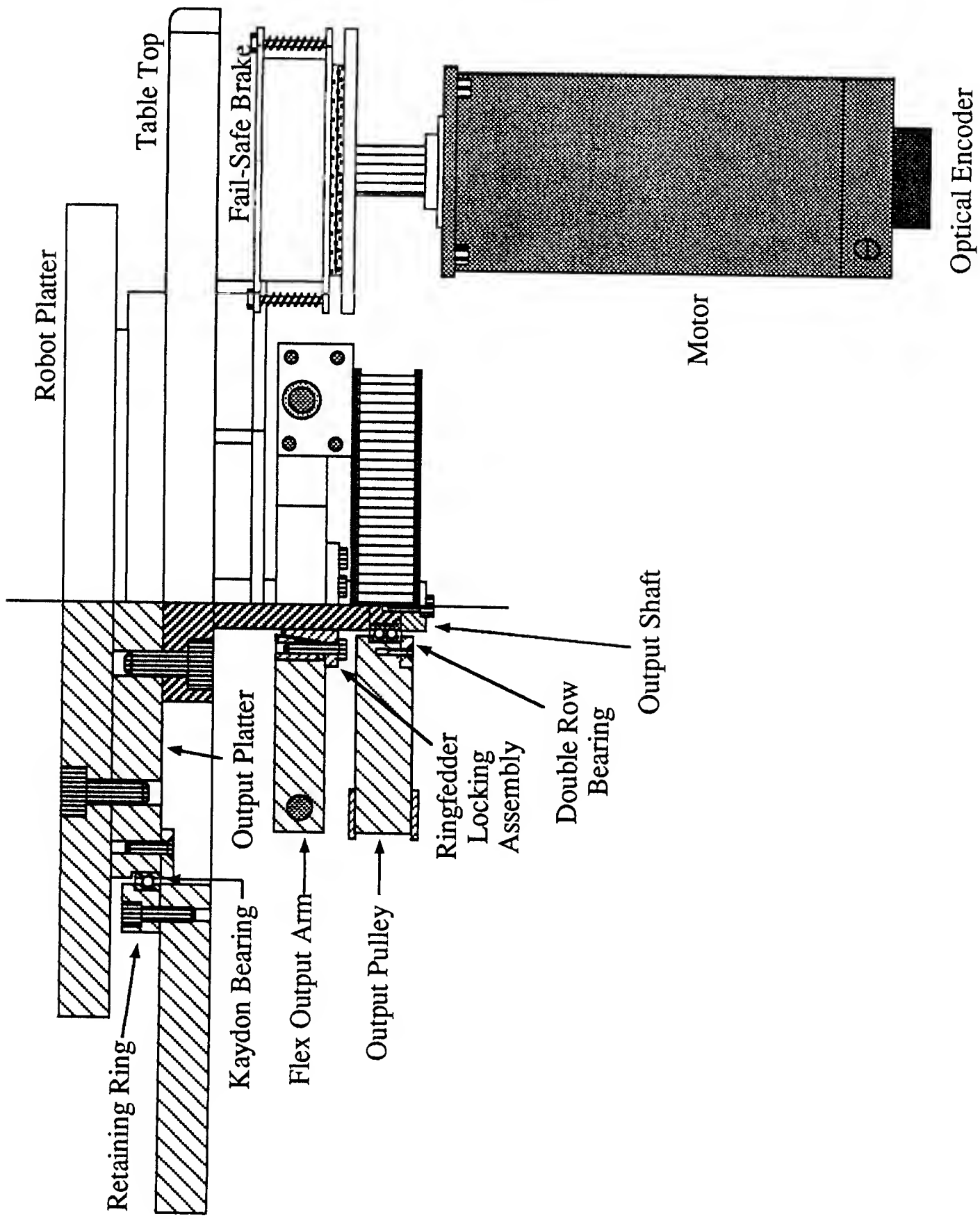


Figure 3.4: Cross-sectional view of the base components. On the left side, a cutaway view shows the general mounting scheme.

the retaining ring of the Kaydon, so the robot platter or other fixtures can be bolted on.

When you are using only the base joint of the robot, the platter can spin freely. When the rest of the robot is bolted on, power and signal cables are wrapped around the shaft and pass through the output platter. To protect these cables, a set of three microswitches (not shown on the drawings) are located under the table. The output platter carries a small aluminum block under the table that trips the microswitches. The middle microswitch is connected to the emergency stop system described in Section 4.2, and the outer two microswitches serve as warnings to the computer that the base is coming near the emergency stop.

3.2.3 First Joint Design

The first joint of the robot is pictured in Figures 3.5, 3.6, and 3.7. It has the same basic motor/pinion/idler/pulley/brake/encoder package as the base, described in Section 3.2.1. The flex output arm of the robot and the large pulley ride on a steel shaft held 15 inches above the base of the robot. This height serves two purposes; first, it provides necessary space for the timing belt between the motor and output pulley. Second, it increases the effective workspace of the robot. Due to laboratory space limitations, the total reach of the robot was limited to 52 inches. Given the size of the robot base, the 15 inch height of the output shaft maximizes the angle that the robot can reach down while still keeping it clear of the ceiling when pointed up.

The flex output arm is pinned and bolted onto the link connector. Each holds a 205 ball bearing (ABEC 7) that ride on the same shaft as the output pulley, as shown in Figure 3.7. This guarantees alignment of the pulley, flex output arm, and output link connector. Keeping the shaft stationary also removed the problem of mounting it properly on a set of bearings.

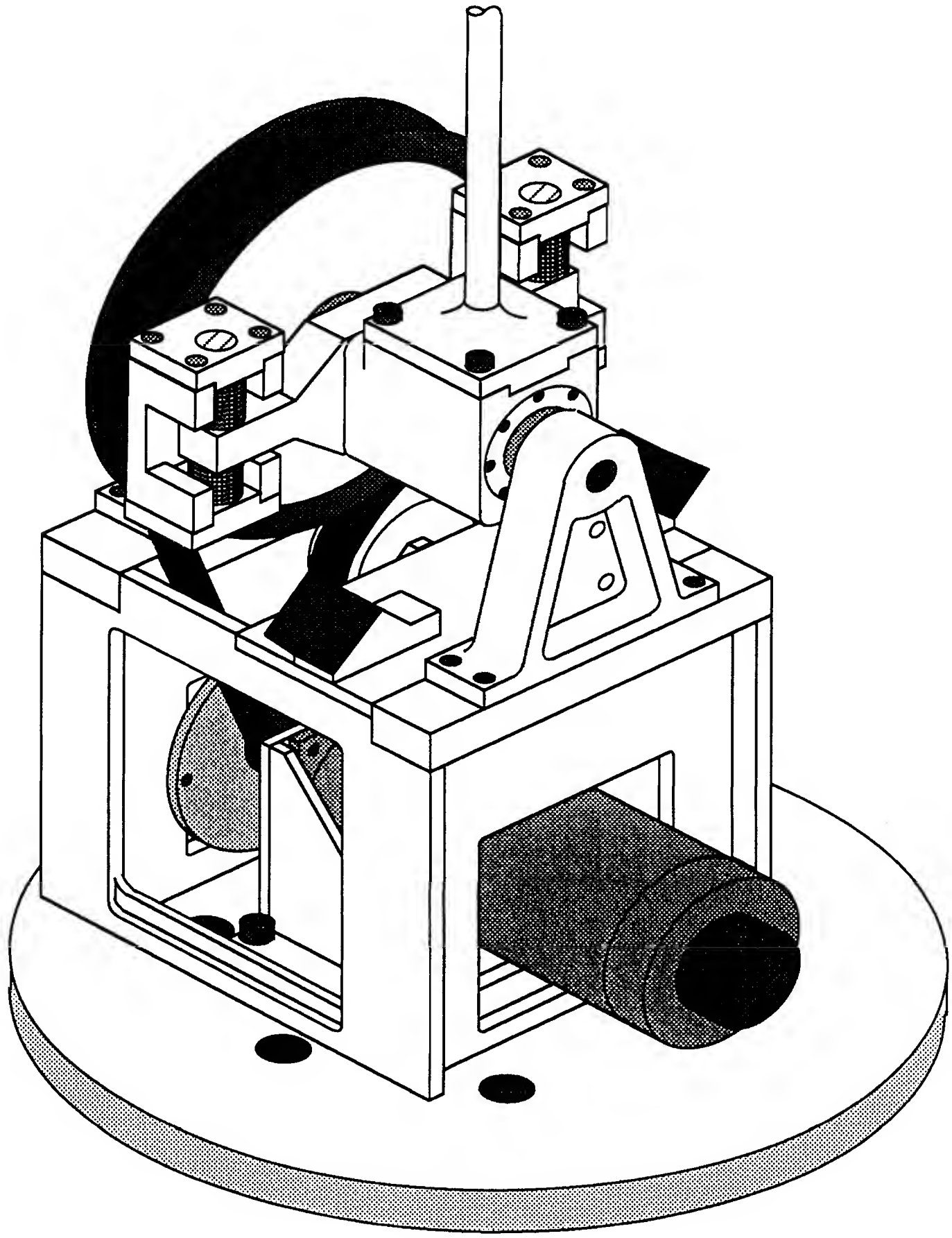


Figure 3.5: Isometric of the first axis.

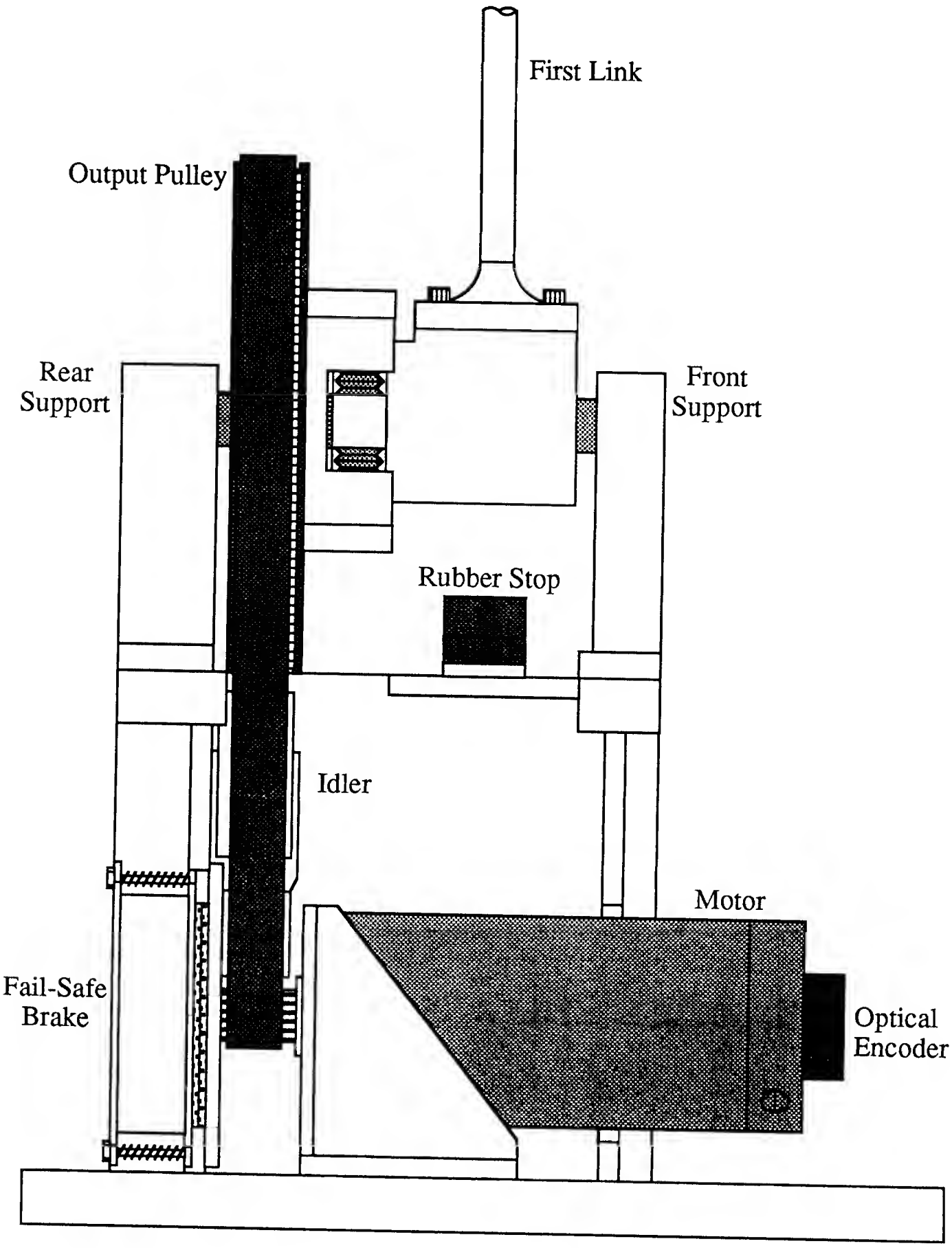


Figure 3.6: Side view of the first axis. Part of the housing has been removed so you can see the motor and brake.

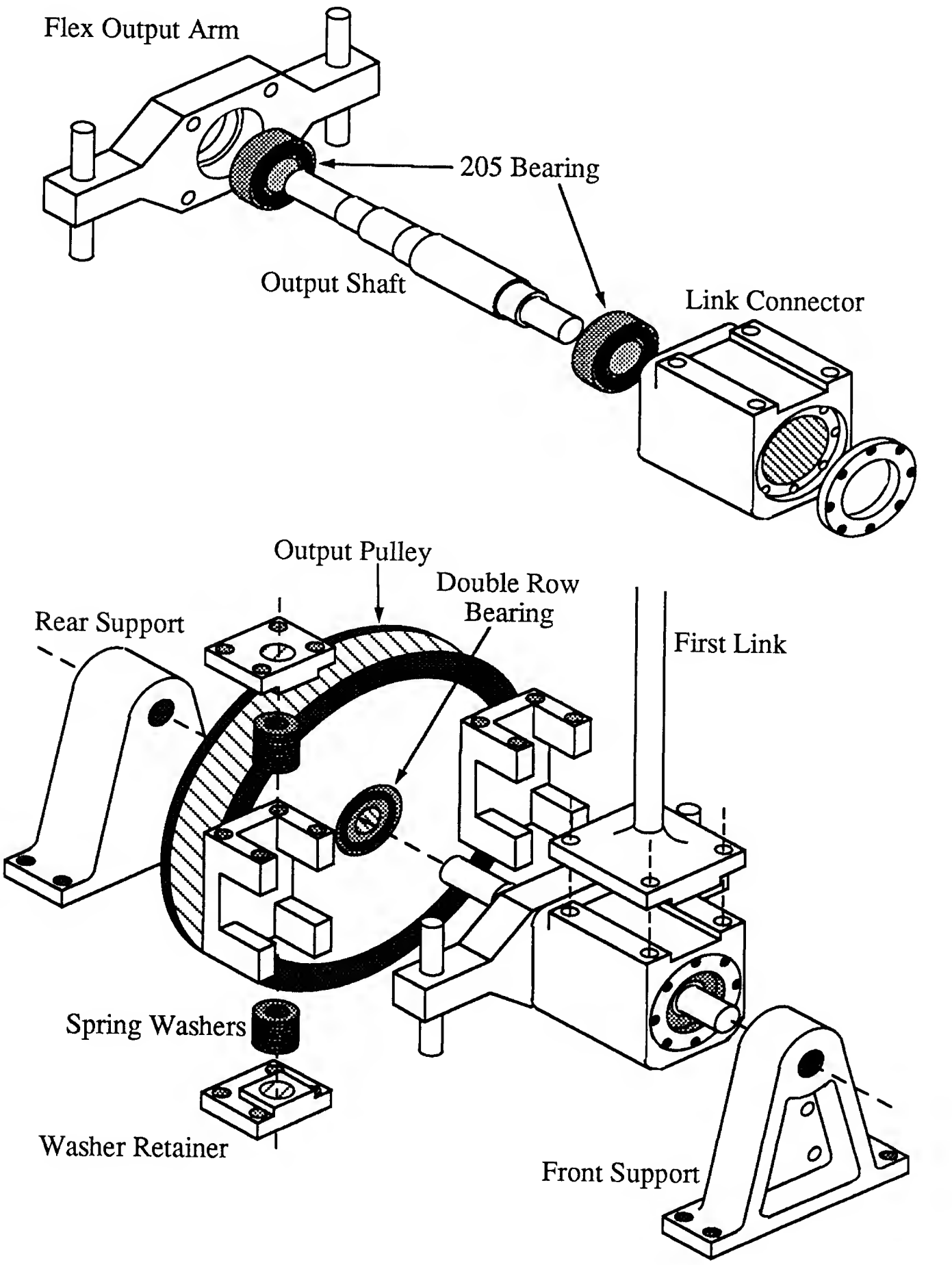


Figure 3.7: Exploded view of the output shaft of the first axis.

The aluminum plate under the joint output is removable and holds a pair of rubber stops (see Figures 3.5 and 3.6) and emergency stop microswitches (not shown). When a different set of links is put on the robot, this plate is replaced with a new one that has rubber stops and microswitches mounted correctly for the shape of the new link. We also have mounts and connectors for additional microswitches that serve as warnings to the computer that the first link is getting near an emergency stop. These are not being used in the current configuration of the robot.

3.3 Elbow Joint Design

The design of the elbow joint was trickier than the design of the first two joints. At the base, weight and size are not too important, so our design ended up optimizing for power and convenience of maintenance. The elbow joint is the opposite: we had to minimize the weight while meeting our power requirements. Please note that keeping the weight down at the elbow is not a design goal based on the available torque at the base; quite the opposite. There is plenty of power at the base. The elbow weight had to be minimized for vibrational reasons, as explained in Chapter 2. We can get the 3 hertz vibratory mode with any weight of elbow joint as long as we size the links appropriately. But the stress level in the links is minimized when the weight carried by the arm is mostly in the payload and the elbow weight is small. For our desired payload of 3 pounds and total arm reach of 52 inches, we set a target stress of 15% of the yield stress of the links when the arm is being deflected by gravity. That constraint translates to an elbow joint weight of 15 pounds or less.

Initial designs considered using a direct drive DC motor or a hydraulic joint. Direct drive was eliminated because commercially available motors in the torque range we wanted weighed 10 pounds. With a brake, encoder, bearings and flexible coupling added on, they would have weighed over 15 pounds. The hydraulic joint was much more promising: a Helac rotary actuator only weighs a few pounds and

has virtually unlimited power. We abandoned this approach after deciding that our laboratory would not be suitable for hydraulic equipment. The design we settled on is a DC torque motor driving a 5:1 planetary gear set. This gave the desired output torque and speed and kept the weight down to a respectable 13 pounds.

3.3.1 Power Train

The gear set and motor were choosen to minimize the amount of time it takes the joint to swing a 3 pound payload at the end of a 22" link through both a 30 degree motion and a 180 degree motion. Because of the rather high torques provided by the motors we examined, the 180 degree motion is limited primarily by the top speed of the motor and the 30 degree motion mainly by the maximum torque. The best motor was the QT-3802 torque motor. For the QT-3802 motor, the best 30 degree move took 0.102 seconds with a 7:1 gear reduction and the best 180 degree move took 0.331 seconds with a 4:1 gear reduction. We compromised on a 5:1 gear reduction with a 0.113 second 30 degree move and a 0.351 second 180 degree move.

The elbow joint is driven by the QT-3802 Inland Motor permanent magnet DC torque motor. The motor is rated at 42.7 volts, 6 amps. The torque constant is 0.800 ft-lb/amp and the back EMF constant is 1.085 volts/(rad/s). This gives a rated output torque of 4.8 ft-lb and a top speed of 39.3 rad/s or 375 rpm. However, the motor has samarium-cobalt magnets, so it is safe to hit it with a little more current and voltage. Our current amplifier can supply 20 amps for 2 seconds or 6 amps continuously at 60 volts. That gives a maximum torque of 16 ft-lbs and a top speed of 55.3 rad/s or 528 rpm. The motor weighs 2.6 pounds.

The motor mounting scheme is shown in Figure 3.8. The rotor is attached to an aluminum hub with Locktite. The aluminum hub is attached to a steel shaft, again with Locktite. The steel shaft is supported by a pair of 103 sealed ball bearings (ABEC 7) that ride inside of the joint housings. The pinion of the planetary gear set

is welded into the end of the motor shaft. The stator ring of the motor mounts to the front housing. The joint housings are of aluminum. To minimize their weight, excess housing material around the bolt holes has been removed. The flattened section at the bottom of the housings is where the link from the first joint is attached.

The planetary gear set fits into the front housing as shown in Figure 3.9. It is a Matex LGU 75–M 5MLG8, 5:1 reduction, steel planetary gear set. The "75" stands for the diameter, 75 mm or just a little under 3 inches. The gear set input is rated at 32.6 ft-lbs of torque. The output shaft is welded to the gear set and is supported by a 5203 double-row sealed ball bearing. This creates a watertight cavity for the gear set, allowing us to lubricate it with either grease or oil. We are currently using grease.

3.3.2 Output of the Elbow Joint

The optical encoder for the elbow joint can be seen on the back side of Figure 3.9. It is an HP HEDS–9000 optical reader and an HEDS–6100 codewheel. The codewheel has 1000 counts, and the reader generates quadrature information, so with the 5:1 gear reduction we get 20,000 counts per revolution of the output. For protection, the encoder is recessed within the rear housing.

The output arm of the robot is supported by a pair of bearings, shown in Figure 3.10. The bearings are Kaydon JU series, 4.5 inch bore, X-type contacts. They are shown inside of their aluminum housings. Each bearing housing has a flat crown which is where the second link is attached to the elbow joint. The rear bearing rides on the rear housing, just outside of the optical encoder. The front bearing rides on the output support—the same piece that holds the double row bearing which supports the planetary gear. This helps guarantee proper alignment of the bearing sets. Also shown at the back of Figure 3.10 is the rear bearing retainer which holds the bearing in place and doubles as a brake mounting bracket.

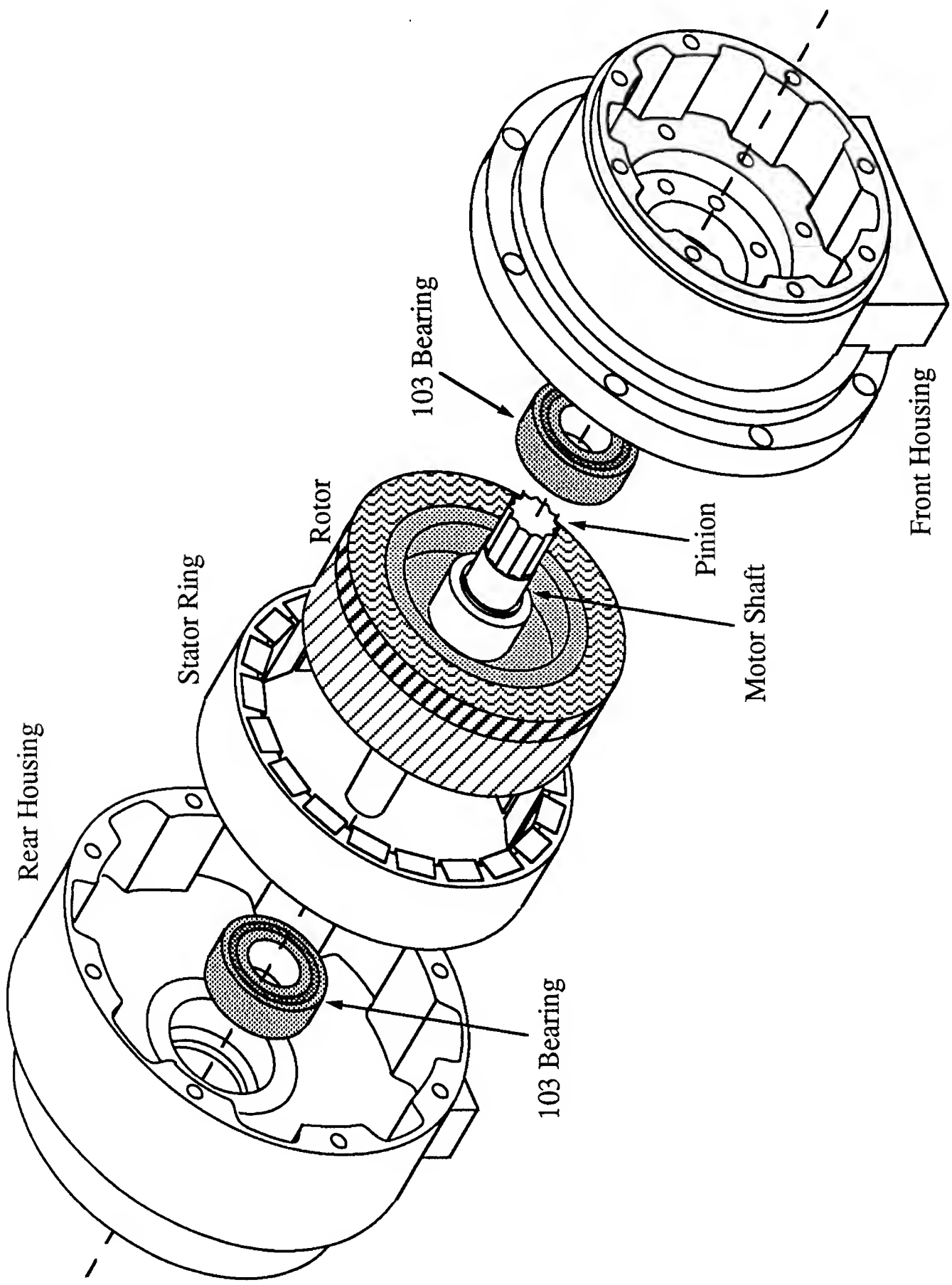


Figure 3.8: Motor assembly for the elbow joint.

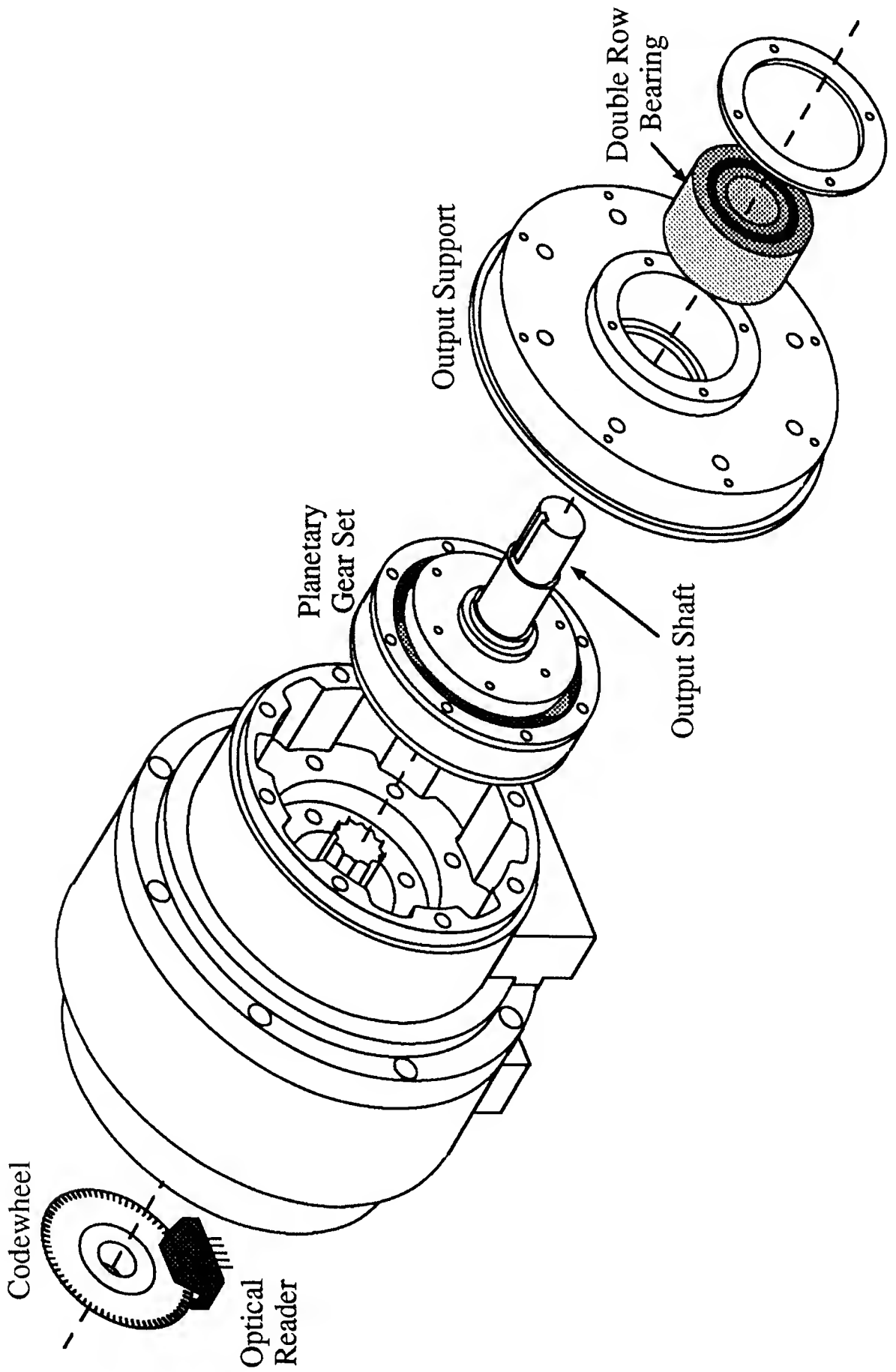


Figure 3.9: Elbow joint gear assembly and optical encoder.

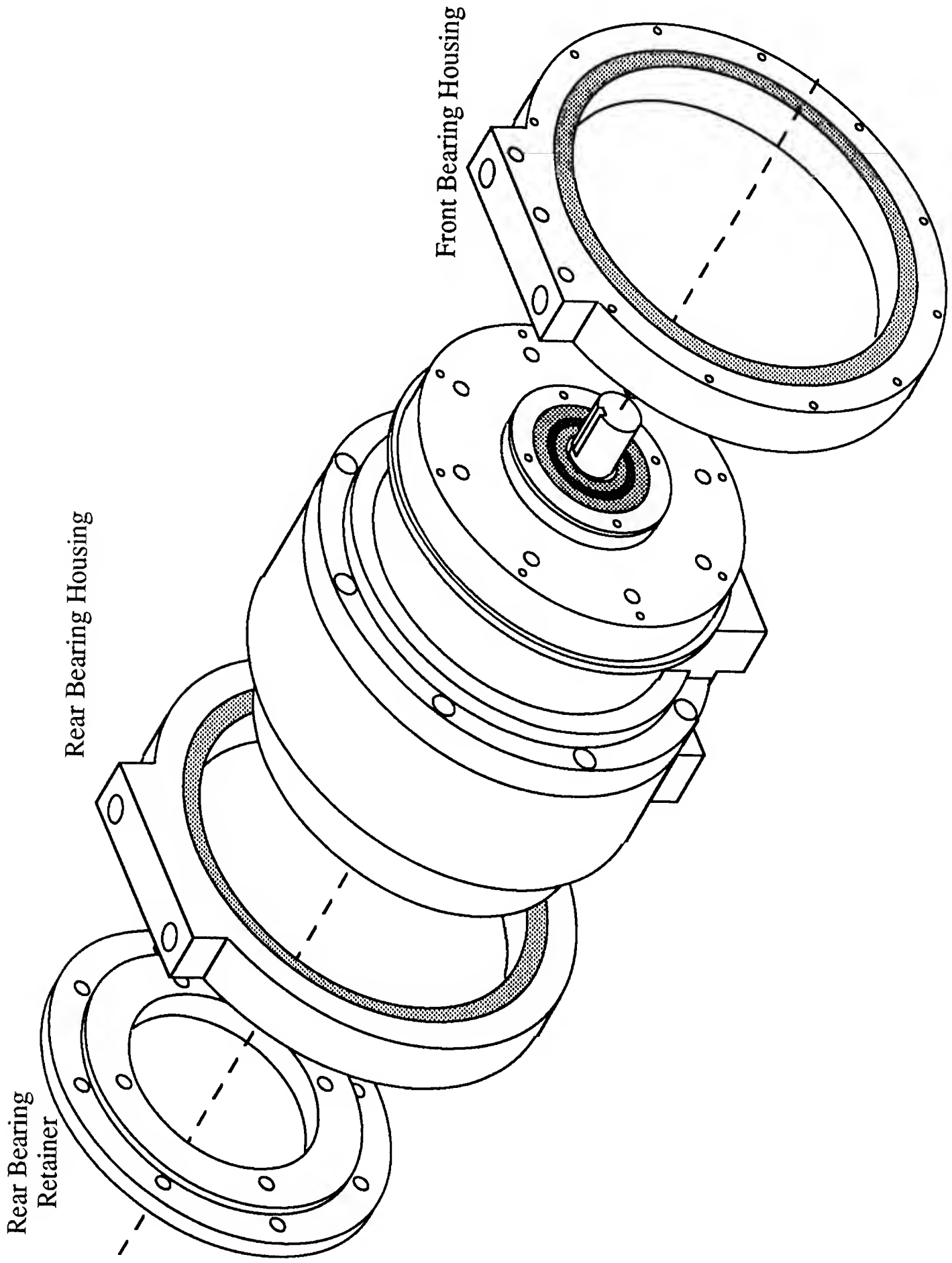


Figure 3.10: Elbow joint output bearings

Figure 3.11 shows the final pieces of the second joint. The output shaft from the planetary gear is coupled to the output arm that holds the spring washers. The arm is held onto the shaft primarily through a friction fit provided by a nut at the bottom of the arm. For protection, the shaft is keyed as well. The arm has two dowel pins which hold the spring washers. The spring washers ride inside of the upper coupling and two steel washer retainers hold them in place.

An Electroid MSFB–26 Fail-Safe brake mounts on the back of the joint and connects directly to the motor shaft. The brake runs on 24 volts/.24 amps, has a rated holding torque (at the output) of 12.5 ft-lbs and weighs 1.4 pounds. The brake is just a smaller version of the fail-safe brakes that are used on the base joints. This particular brake had some problems with releasing correctly, so we replaced the springs with a softer set. The current holding torque of the brake is closer to 8 ft-lbs.

The elbow joint is powered by a Copley Controls Corporation Model 215 Pulse Width Modulated switching amplifier. The 215 switches at 22 kHz and takes an input voltage of 15 to 80 VDC. A stepdown transformer with a rectifier/capacitor circuit that is also manufactured by Copley Controls supplies 60 volts of DC power. The amplifier bandwidth is rated at -3 dB @ 1000 Hz and the slew rate is 130 amp/ms. A model MB2 mating test board attached to the amplifier provides adjustable current limits (1–20 amps), gain, and balance. The 215 can supply 20 amps for 2 seconds and 6 amps continously. It is controlled by a differential ± 10 volt signal and has an inhibit line to stop the amplifier when pulled low. The 215 has a reliable overload protection circuit with a reset switch and indicator lights to show the amplifier status.

The emergency stop microswitches and rubber stops are not shown in these pictures. They are mounted directly to the first link of the robot rather than to the elbow joint. This is done because each set of links requires a slightly different configuration for the switches and the rubber stops. The emergency stops are a pair of microswitches that ride against the front bearing housing and detect the crown. The

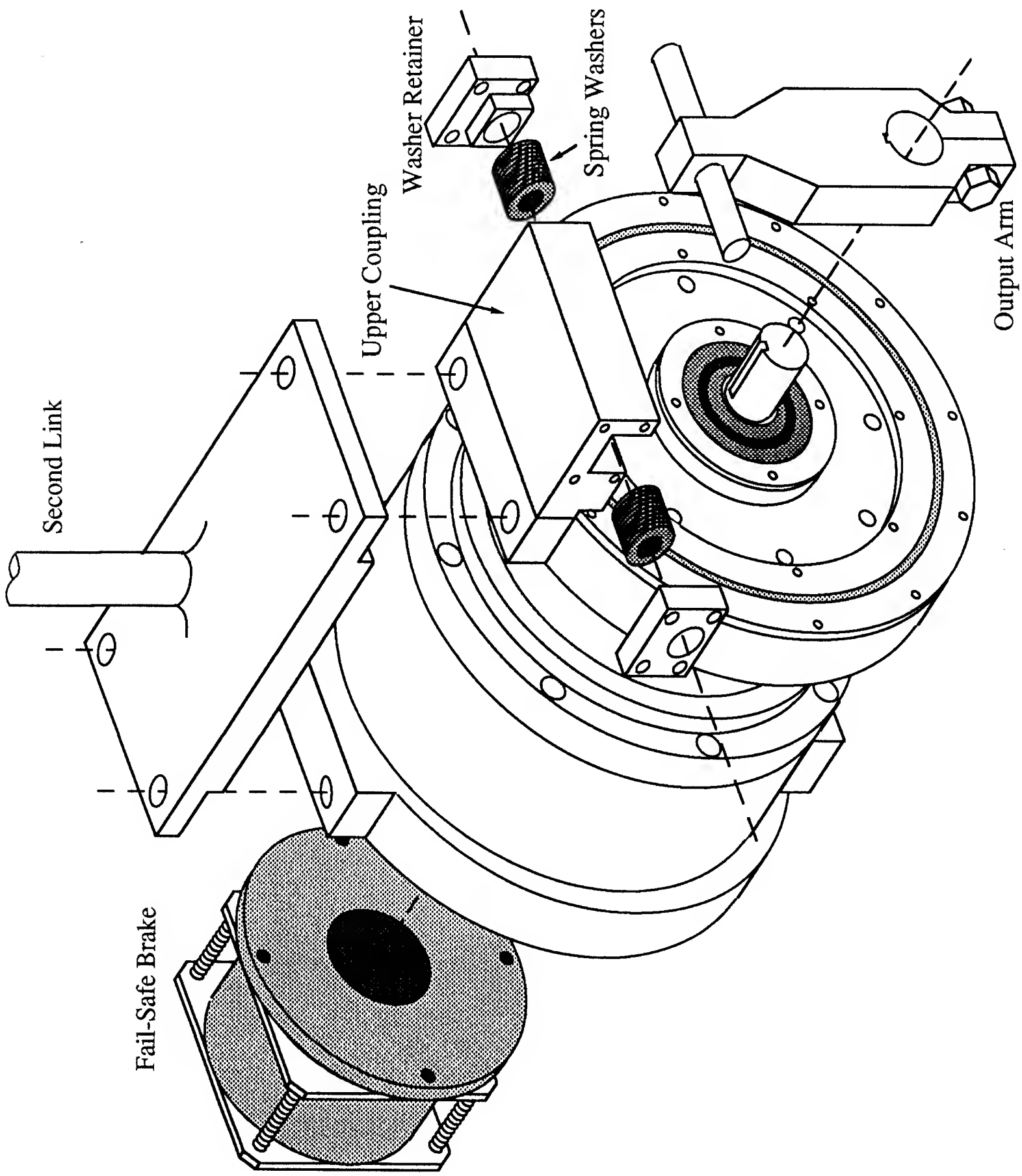


Figure 3.11: Second joint flexible coupling and fail-safe brake.

rubber stops fit between the first link and the second link and prevent them from crashing together.

3.4 Link Design

The conclusion of Chapter 2 includes recommendations for the best materials to use in the links of a flexible robot and what they their shape should be. In our case, we decided to use steel links. Although Section 2.3.2 explains why a good 2014 aluminum alloy is stronger than steel, we ended up using type O1 oil-hardening drill rod as our link material. There are several reasons for this choice. First, for this particular design the weight of the links was negligible compared to the weight of the payload or the elbow joint. Although aluminum makes a lighter link than steel, this turned out not to be a factor. Second, we designed a mounting scheme for the steel links where they are welded into a gently curving endplate. This forms a strong joint with a minimal stress concentration. Third, the drill rod is inexpensive and commercially available in a multitude of standard sizes (to the 1/64th of an inch) which makes it reasonable to keep many spares on hand. Fourth, the steel can be hardened. The strength calculations in Section 2.3.2 did not deal with the possibility of hardening a link. The extra toughness gained offsets the strength advantage of the aluminum. And finally, steel vibrates very well. Internal damping is minimal.

For vibration testing we use a set of steel links. Overall link configuration appears in Figure 3.1, although a good side view of what the bottom of a link looks like can be seen in Figure 3.6. Each link consists of a piece of oil hardening drill rod silver-soldered into link holders on either end. The link holders are a flat steel plate with a raised hole to accept the link. The raised hole strongly resembles a volcano and was cut using a endmill with a round end. By silver-soldering the link in place and then sanding the joint smooth we minimize the stress concentration.

The first link has a diameter of 5/8" and an overall length of 20 inches. The

vibratory part of the link is 18.5 inches long. The second link has a diameter of 5/16", an overall length of 21.5 inches, and a vibrating length of 18.5 inches. The end of the second link is welded to a one inch diameter, 3 inch long piece of steel. This slug forms a permanent payload of one pound. An aluminum cylinder, visible in Figure 3.1, fits over this slug and fastens on with a single bolt. This raises the payload to 3 pounds and makes it easy to add or remove weight.

We do have an additional collection of links, the most commonly used being the aluminum "training" links. They are the same length as the steel links but are made out of aluminum bar stock and are a great deal stiffer than the steel links. They are used primarily for debugging computer code.

3.5 Performance

To wrap up this chapter, I'd like to list the current hardware specifications for the robot, shown in Table 3.1. The acceleration data for the base assumes that the arm is fully extended. When the elbow joint is folded up, the inertia drops to roughly half and the acceleration improves accordingly.

Vibrationally, the robot behaves as designed. With the flexible links in place and the joints locked down, the arm has a measured lowest natural frequency of vibration of 3.0 Hertz and a second mode of vibration of 8.0 Hertz. This agrees well with the measured static deflection under gravity at full extension of 1.3 inches. These frequencies drop when the arm is actively controlled by the computer and we add the joint flexibility. Section 5.2 goes into more detail on how the arm vibrates.

I'll conclude this chapter with some information on how the joint flexibility will affect the natural frequency of vibration and what range of joint flexibility is available. Table 3.2 shows the current operating range of the spring washers. The static deflection is the deflection of the end of the robot under gravity when it is fully outstretched. As Section 2.2 showed, the endpoint deflection under gravity directly

Lengths	Base to Elbow	25	inches
	Elbow to Tip	26	inches
	Overall	51	inches
Weights	Elbow Joint	13	pounds
	Payload	3	pounds
Base Joints	Maximum Torque	240	ft-lb
	Continuous Torque	45	ft-lb
	Maximum Speed	240	RPM
	Encoder Resolution	0.009	degrees
Elbow Joint	Maximum Torque	80	ft-lb
	Continuous Torque	24	ft-lb
	Maximum Speed	106	RPM
	Encoder Resolution	0.018	degrees
Base Motion (with arm extended)	Inertia against Base	140	ft^2-lb
	Maximum Acceleration	62.3	rad/sec^2
	Acceleration at the tip	8.22	g
	Maximum tip speed	106.8	ft/sec
Elbow Motion	Inertia against Elbow	15	ft^2-lb
	Maximum Acceleration	165	rad/sec^2
	Acceleration at the tip	10.7	g
	Maximum tip speed	24.1	ft/sec

Table 3.1: Robot specs with steel vibratory links in place.

Base Joints	Minimum Spring Constant	37,500	in-lb/rad
	Static Deflection	0.60	inches
	Nominal Spring Constant	300,000	in-lb/rad
	Static Deflection	0.075	inches
Elbow Joint	Minimum Spring Constant	5,700	in-lb/rad
	Static Deflection	0.26	inches
	Nominal Spring Constant	20,000	in-lb/rad
	Static Deflection	0.075	inches

Table 3.2: Spring washer ranges

correlates with the lowest natural frequency of vibration. The nominal spring constant is what we designed the normal flexibility of the joint to be. This is with stacks of 12 to 15 springs on each dowel pin. We can raise this spring constant almost arbitrarily high by using fewer springs. But the spring constant cannot be pushed arbitrarily low. The lowest possible stiffness is achieved by using extremely thin spring washers and outfitting the joint with as many as will fit onto the dowel pins. To get the stiffness still lower, you also can use only two of the four dowel pins on each of the two base joints. The minimum spring constant obtainable by these methods is shown Table 3.2. If you calculate the behavior of a robot with stiff links and the minimum spring constant, you still have a natural frequency of 3.4 Hertz. In fact, by using slit spring washers or simple compression springs, this number could be lowered still further.

Electronics and Software

Chapter 4

I learned the hard way that the design and construction of a robot takes just the first half of your time. The second half is putting together the electronics, wiring the robot, assembling a working computer system and writing software. This chapter deals with the second half of the robot.

The Flexbot as described in Chapter 3 is a free-standing robot, but is missing some basic ingredients. It needs wiring for the amplifiers and encoders, power for the brakes, a plan to handle starting and stopping the robot, an emergency stop, a method of teleoperation, and a computer system that can control all of the above. This chapter describes these components. It is broken up into five sections: the computer hardware, operator controls to safely start and stop the robot, general electrical circuits/wiring, servo loop design/inverse kinematics, and the computer software that runs the robot.

4.1 Computer Hardware

To provide adequate computational speed for control of the robot we use a simple parallel computer. Three separate processors run simultaneously and divide the work of controlling the robot. They share a common backplane which has a complement of interface cards to connect to the outside world. A separate computer provides a development environment and data storage.

The computer system we have assembled is not unique; there are four other systems similar to it our laboratory. This makes it easy for us to borrow components or test hardware on working systems. The computer system is comprised of a Sun 3/180 Unix Workstation connected to a VMEbus expansion box. The expansion box holds a system controller, three single board processors, a digital to analog converter board, an analog to digital converter board, five optical encoder reading cards, a digital I/O board, and extra memory. The backplanes of the Sun and the VMEbus expansion box are connected together so that we can transfer data to and from the Unix system at high speed. The VMEbus box and Sun are mounted in the same rack; a picture of that rack is shown in Figure 4.1.

Figure 4.2 shows the boards in the expansion box. The Sun 3/180 is used solely as a convenient development environment, it does not deal with any of the robot control because Unix is not a real-time system. All robot control is done by the single-board processors located on the VMEbus. The software for the Sun (described in Section 4.5) provides a simple user interface to the processor boards, runs the software compiler, and provides space for permanent data storage.

Here is a short summary of each of the boards in the VME box; a description of the board's ports, capabilities, and the use of the board in controlling the robot. The boards are listed from left to right as they appear in Figure 4.2. The VME box is made by Electronic Solutions and contains the standard VME backplane, power and reset buttons, and a rather hefty power supply that drives all of the VME cards, the joystick, and some other circuitry.

Ironics IV–3273 System Controller:

All VMEbus systems require a board that mediates bus access and provides functions such as a system reset and clocking. We purchased this model for compatibility with our processor boards. The controller does have some capabilities that we may use in the future which include a SCSI port, two parallel

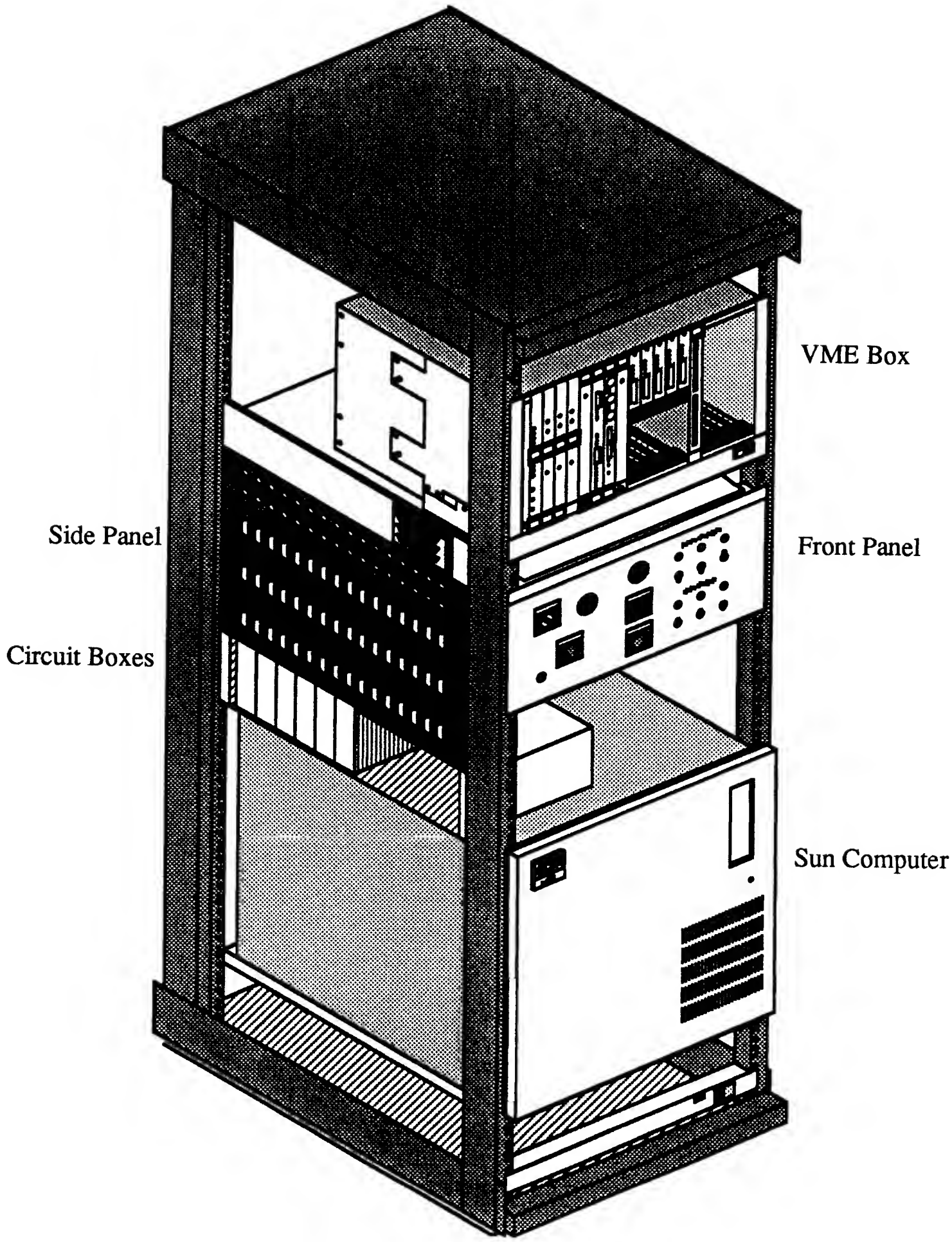


Figure 4.1: Mounting rack and components for the computer system.

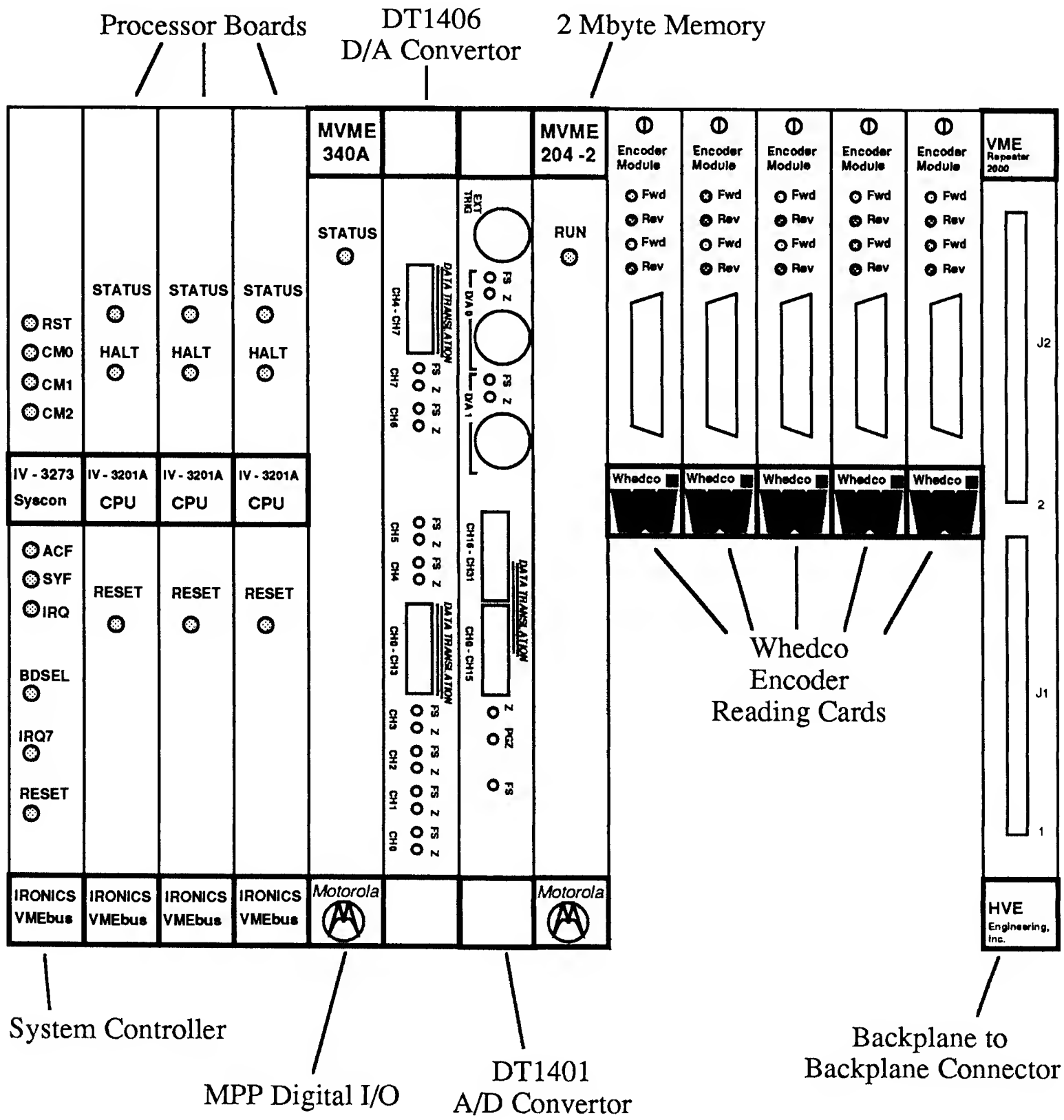


Figure 4.2: VMEbus boards in the expansion box.

ports, and an extra serial port.

Ironics IV–3201A Processor:

A single board, self-contained microcomputer. It is based on a 16 MHz Motorola 68020 processor with a 68881 floating point coprocessor and one megabyte of dynamic RAM. The Ironics processor card has several nice features, including a mailbox for interrupt driven communication between processor cards and dual-ported RAM that allows one processor card to access the memory of another processor card without interrupting it. Our control system has three of these boards (software described in Section 4.5).

Motorola MVME 340A Parallel Interface/Timer Module:

The MVME board provides digital I/O for the robot system. It has 50 I/O lines that can be used independently or in blocks of either 8, 16 or 32 for parallel data transfer. In addition, there are 8 lines for handshaking, 6 lines for timing functions and three 24 bit timers. We use this board to read microswitches, sense if the amplifiers are enabled, and to get information off of the joystick box.

Data Translation DT1401 Interface Card:

The DT1401 is a 12 bit A/D converter. It has 32 channels of A/D which can either be used as 32 single-ended channels (meaning they all have a common ground) or as 16 differential channels. The input range is either 0 to 10 volts or ± 10 volts. However, the board can prescale the input by a factor of 1, 2, 4, or 8 which gives it an effective range of either 1.25, 2.50, 5.00, or 10.0 volts. Each channel takes 10 microseconds to sample and 15 microseconds to convert a reading which gives an overall sampling frequency of 40 kHz. Additionally, the DT1401 has two 12 bit D/A converters that can be configured to be either unipolar or bipolar, 5 or 10 volts maximum. And the DT1401 has 16 lines of

digital I/O. We currently use the DT1401 for reading the joystick position. It also can be set up to read the current feedback from the amplifiers.

Data Translation DT1406 DAC Card:

The DT1406 is a 12 bit D/A converter. It has 8 channels of D/A which can be set for either 0 to 10 volt operation, or −10 to +10 volt operation. The DT1406 is used to send control signals to the amplifiers.

Motorola Memory Board:

This board contains two megabytes of dual ported RAM that is memory mapped onto the VMEbus. The memory available on the Ironics boards is already adequate for robot control; this memory exists to facilitate communications with the Sun. It provides a convenient place to store large quantities of data.

Whedco Dual Channel Incremental Encoder Interface Card:

Each Whedco board provides two channels of encoder interface. That is, each card controls two optical encoders. The board accepts single ended or differential signals, can be configured to provide power to the optical encoders at 5 or 12 volts, and can be set for 1, 2, or 4 counts per line on the encoder. Position tracking is 32 bit, either 0 to 4,294,967,295 or ± 2,147,483,648 counts, which means that we can run the base axis at full speed for 3 hours and 43 minutes before the counter overflows. Each channel has four set points that control a digital output line. They can be independently programmed to be activated when the position counters are less than a set point, greater than a set point, or when they are within a given range. These cards are a convenient way to read the optical encoders in the robot. We own five of them because we are planning to use them on a different robot in our laboratory (which has 7 axes) and because we are the only group in the laboratory who has this type of VMEbus

card. All the other cards we use in our system are duplicated by other research groups, so we have a convenient source of spares or temporary loans.

HVE Engineering VMEbus Repeater 2000:

The HVE card actually is one of a pair: one card sits in the VME box and one card sits inside of the Sun. These two cards and a pair of shielded ribbon cables connect the backplane of the Sun to the backplane of the VME box and allow normal memory access between the buses. A modification to the Unix kernel on the Sun maps the memory addresses of the cards in the VMEbus to an area outside of the normal RAM or virtual memory used by the Sun.

4.2 Operator Controls

The operator controls the Flexbot by the "Front Panel". The Front Panel is an array of switches, buttons and lights designed to allow the operator to selectively run the robot with fewer than all three axes, enable/disable the amplifiers, and control the brakes. Figure 4.3 shows the layout of the panel. Its position on the card rack can be seen in Figure 4.1.

The front panel provides a power on/off keyswitch, a set of running lights to indicate the robot state, switches to turn on and off the amplifiers and brakes at each axis, and a set of emergency stop panic buttons. Figure 4.4 is the ladder logic diagram for how the controls behave. I apologize for the diagram; although ladder logic is a common (albeit old) method to display relay circuitry, many people I know have never seen it. It is very simple to read. Switches and buttons are represented by two small circles and a line connecting them. The style of switch can be inferred from the line: for example, the emergency stops are mushroom head buttons, the start button is a normally open button and each axis enable is a double pole toggle switch. The circle with the letter "R" represents a relay; that is, a relay closes when power is

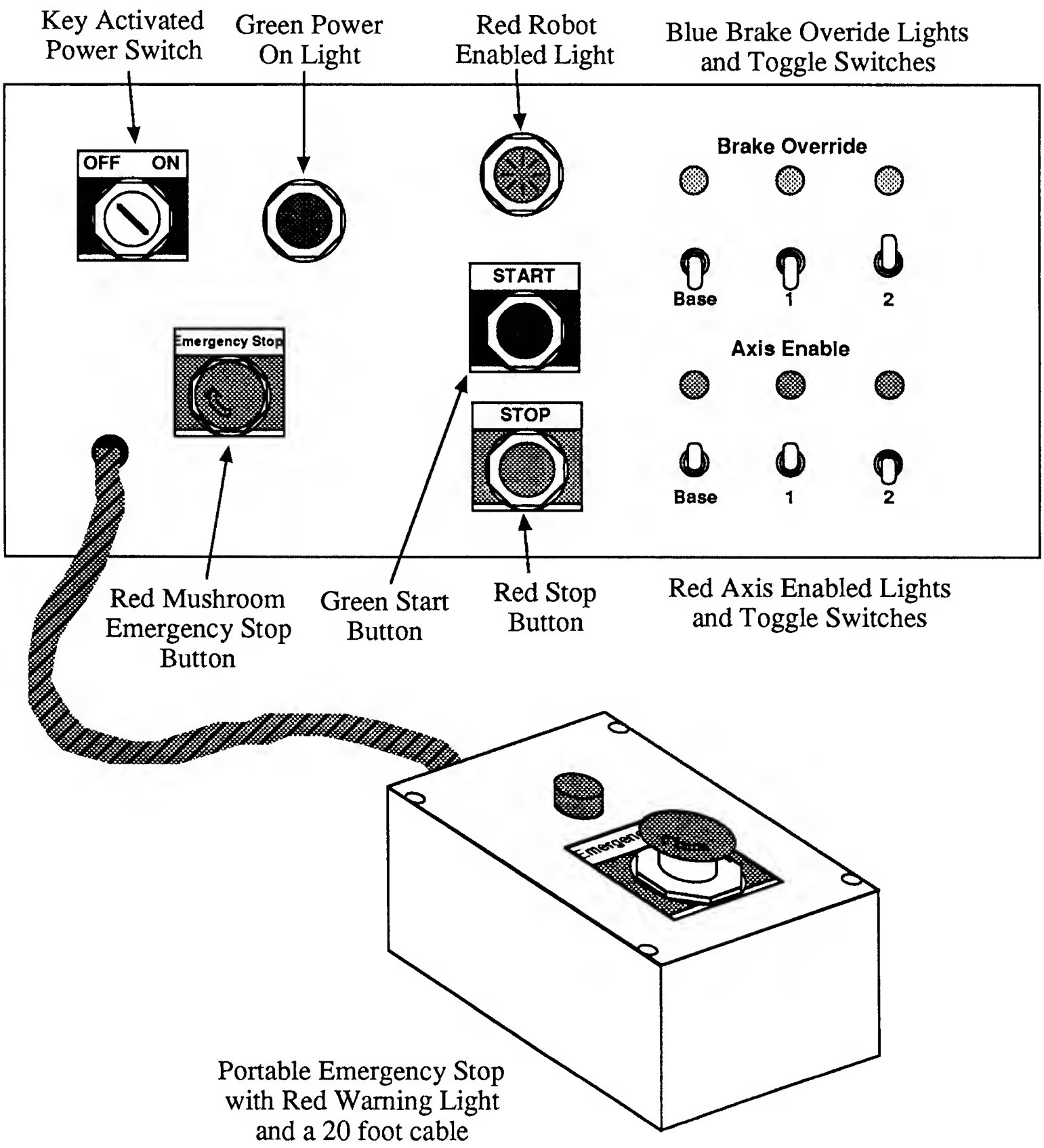


Figure 4.3: Operator controls: The Flexbot front panel.

flowing through the switches to the left of the circle. Relay contacts are represented by two parallel lines and are labeled with the relay number. Power always flows from left to right.

The behavior of the front panel is as follows: Initially all power to the system is off. That means that the amplifiers are disabled and the fail-safe brakes are closed. Turning the key in the keyswitch turns on the green "Power On" light and allows AC current to reach the 28 volt DC power supply. The 28 volt DC power supply provides power to the brakes, the front panel lights (except for the green "Power On" light), and the amplifier enable relays located in the base of the robot.

The R1 relay switches DC power for most of the system. It is enabled only when all of the emergency stop switches are closed; emergency stops are all normally closed switches. The front panel has two emergency stops. One is mounted on the front panel and one is mounted in a portable box and attaches to the front panel with a heavy 20 foot coiled microphone cable, as shown in Figure 4.3. Both of these emergency stops are red mushroom heads and they lock when pushed shut. They release when the button is twisted.

In addition to the two emergency stops at the front panel, there are also emergency stops in the robot. Each axis of the robot has microswitches daisy-chained together to form an emergency stop loop. The microswitches are wired normally closed, so if any microswitch of this loop is contacted by a link that has moved too far, the R1 relay opens. This also holds true if one of the microswitch cables gets unplugged by accident, which insures that a careless operator has not forgotten to connect the emergency stops. Unlike the mushroom head buttons, the microswitches are not of the lock-open type. This isn't a problem because once the R1 relay opens the amplifiers will not be enabled until the start button is pushed.

The actual enabling of each amplifier and release of each brake is controlled by the start/stop buttons and the axis enable switches. Normal operation using all three

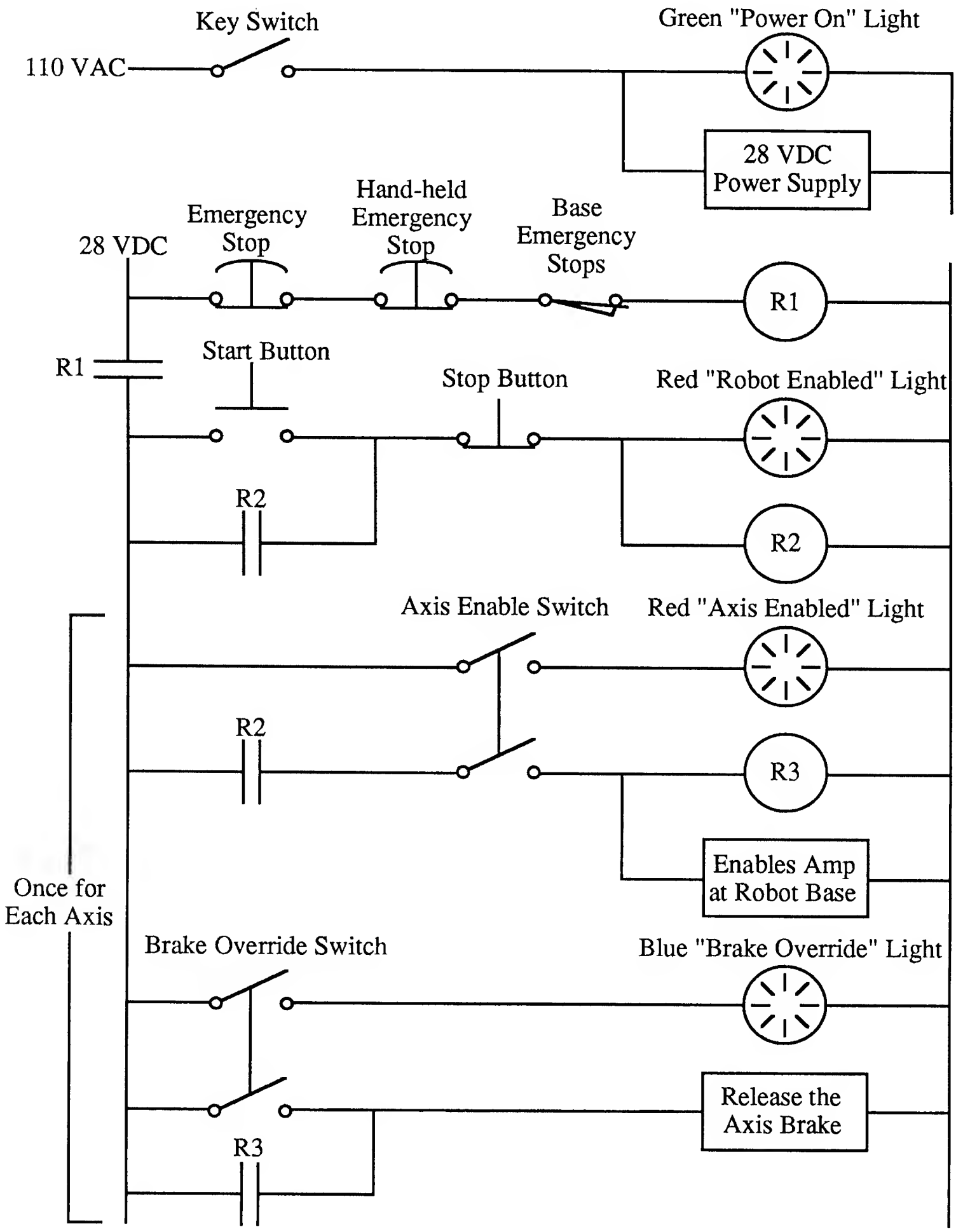


Figure 4.4: Control logic for the front panel of the Flexbot

joints is to flip each axis enable switch up (which turns on the corresponding red "Axis Enabled" light) and push the green start button. The start button turns on the large red "Robot Enabled" light and activates a second relay, R2. The combination of R2 and the axis enable switch directs current to a relay in the robot base which controls the amplifier for that particular axis. It also turns on one of the R3 relays; to save space, Figure 4.4 only shows the circuitry for a single axis. R3 allows current from the 28 volt DC power supply to flow through the brake for that axis. Hence pushing the start button with an axis enable switch closed causes the brake to release and the amplifier to start at the same time.

The R2 relay latches on when the start button has been pushed. The stop button interrupts this circuit and shuts off the R2 relay, and hence turns off all of the amplifiers and closes all of the brakes. Finally, the brake override switches release individual brakes (as long as no emergency stop is closed and the power is on) so that the robot may be repositioned manually without enabling the amplifiers. In normal operation the brake overrides are left off.

One question that has been asked a number of times is "Why bother with the stop button?" After all, pushing an emergency stop button interrupts power to the R2 relay and effectively gives the same result as the stop button. Besides the aesthetic quality, it exists because the functions of stop and emergency stop are not necessarily the same. Emergency stops are a panic device and kill all power to the system, whereas Stop just means turn off the robot. In the future it is conceivable that some additional circuits will be added to the robot (perhaps a manipulator or hand) that might want the emergency stop to kill power, but the stop button to leave power on.

For completeness, let me run through a quick scenario of how the buttons are typically used. The robot has been plugged in and everything is ready to run. The operator turns the keyswitch on, flips all three axis enable switches up (the "Axis

Enabled" lights glow red) and fires up the software for the robot. When the computer is ready to begin servoing the robot, the operator pushes the green start button. The red "Robot Enabled" light comes on, all three brakes click as they release the joints and the amplifiers turn on. The robot is now running. A few minutes later something goes wrong and the operator hits the hand-held emergency stop. The amplifiers shut off and the brakes hold the robot in place. To reposition the robot in the home or upright position, the operator flips up the three brake override switches. The only light on is the green "Power On" light; all others derive their power from the 28 volt DC power supply which is still turned off by the emergency stop. The operator walks over to the robot carrying the closed emergency stop, grabs hold of the robot, and releases the emergency stop. The blue "Brake Override" lights and the red "Axis Enabled" lights turn on. Since the start button has not been pushed the amplifiers are still off, but the brakes release. The operator moves the robot back to the home position and then shuts off the three brake override switches. The robot is now ready to run again.

4.3 Electrical Circuits

This section deals with the remaining electrical circuits in the Flexbot. They consist of the side panel interface with the VMEbus, some general notes on cabling and power, circuits for the optical encoders, the joystick, microswitches, and a protection circuit. It is beyond the scope of this thesis to include specific wiring diagrams for each circuit (and I suspect that few people would be interested in them). Hence this chapter is just an overview of the general circuits installed in the robot and their purposes.

4.3.1 Side Panel

Connections to VMEbus cards are made either through connectors on the front of the cards or by using the spare set of pins on the J2 socket of the VME backplane. This creates a tangle of cables draping down from the VME box and makes changing the wiring of your robot a nuisance. Since we wanted to use this system for controlling other robots, we built an interface panel for the VME cards. Figure 4.5 shows the interface panel, hereafter referred to as the "side panel" to distinguish it from the "front panel" which the operator uses to control the robot (see Figure 4.1).

The side panel has 32 male BNC connectors on the top, 32 female 15 pin D connectors below, and a few random connectors from the system controller card. We deliberately bought more connectors than we needed so that future expansions of the system are convenient. Connections to each of the VMEbus cards are attached to this panel by shielded cables running underneath the VME box. The D/A converters get their own BNC connectors, A/D ports are ganged up 2 to a 15 pin connector and the digital I/O is lumped in groups of 8 channels each so they can be used as parallel ports. The "Robot Axis" 15 pin connectors are special. We wanted to be able to carry all of the return logic from a joint of the robot on one convenient cable. Each "Robot Axis" connector carries one encoder channel, 2 digital I/O lines and an A/D. Finally, the 15 pin connector on the bottom right of the side panel carries the power lines for the brakes, amplifier enable lines, and the emergency stop loop.

4.3.2 Cabling and general wiring

This section provides some general information on the power and signal lines used in the robot. The computer runs off of a 20 amp, 110 VAC line. The base is on a separate pair of 30 amp, 110 VAC lines. The base power lines run into a pair of Signal Transformer Model DU–5 isolation transformers. The first transformer supplies power to the base joint and to a 5 volt DC power supply. The second

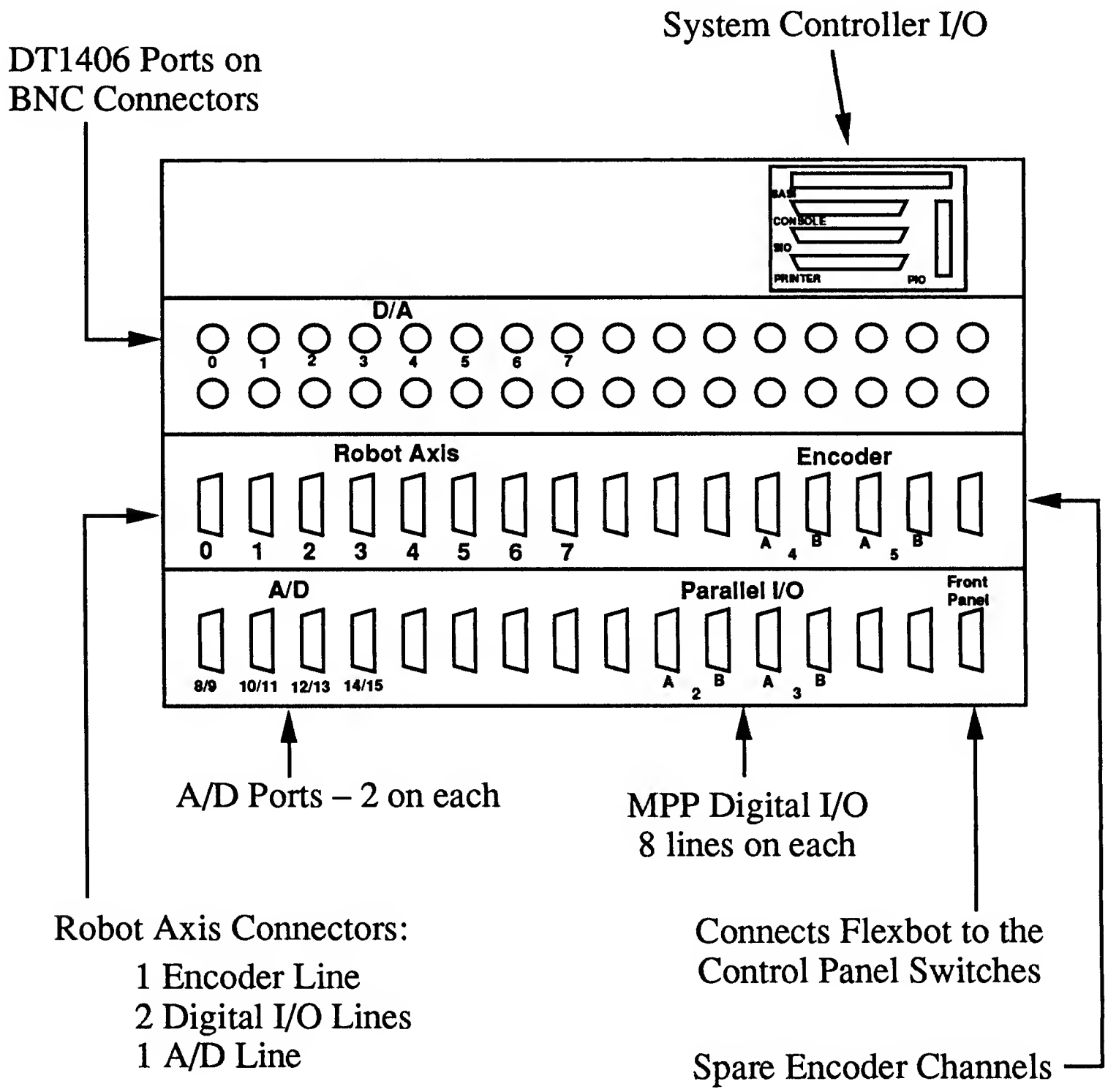


Figure 4.5: Side Panel: Interface for the VMEbus cards

transformer supplies power to the first joint and second joint. The base joint and first joint of the robot get their DC power from rectifier/capacitor circuits connected to the low taps of the transformers, giving them each a little over 100 volts DC. The elbow joint has a step down transformer running into a rectifier/capacitor circuit that supplies it with 60 volts DC.

We deliberately kept the robot and the computer system isolated from each other to keep the switching noise to a minimum. The ground at the robot base floats with respect to the computer. We can do that because the only signal lines currently connecting the robot base with the computer are the D/A signals and the encoder signals. The amplifiers take a floating relative input and the Whedco encoder lines are optoisolated from the base as described in Section 4.3.3. Shields on the signal lines are connected to the computer end only.

The robot requires some special wiring to get power and signal lines from the base to the upper two axes. We didn't want to include a huge slip-ring package, so we opted for cables that travel through the output platter of the base joint. All of the cables from the upper two joints are joined together into a cable package enclosed in Expando sleeving. The cable package passes through a large hole in the base of the first joint, through the output platter, and then wraps around the shaft of the base joint about 6 times. The cable is held off of the flex coupling and the timing belt by a flat aluminum plate. After wrapping around the shaft, the individual cables are separated. They attach to a set of breakaway connectors, just in case the base tries to turn too far.

4.3.3 Optical Encoders

The optical encoders merit a little attention because of the problems we have had with them. The Whedco encoder reading boards do a good job keeping track of pulses, but they require a relatively clean electrical signal. With the PWM amplifiers in the

base of the robot and the 20 feet of cabling required to bring the signal back to the computer they pick up enough noise to make readings inaccurate. The base axis can still function because its cable leaves the base almost immediately, so it doesn't get much noise. The cables for the first and second joint travel around inside of the base quite a distance so they had to be cleaned up before the Whedco board would read them properly.

We implemented a two step process to take care of the noise. A 5 volt DC power supply inside of the base provides power to a buffer box which is mounted on the first joint. The optical encoders of the first and second joint are driven off of this power supply. The return signals from the encoders are fed into the buffer box and into a 74HCT244 line driver. The line driver provides a high impendence signal which is sent through the base via the same cable that supplied the power. This cable is then split and runs across the floor to the computer box. Measuring the signal at the computer end shows the 5 volt signal carrying approximately 2.5 volts peak of noise.

At the computer the encoder signal runs into a conditioning box located in the circuit box section of the rack (see Figure 4.1). The conditioning box contains an HP 2631 Optoisolator (and a few capacitors) powered off of the 5 volt supply in the VME box. The optoisolator guarantees that the robot base and the computer do not have a common ground. The output from the optoisolator is free from noise. In fact, it is so good that the Whedco boards can't handle it. The optoisolators slam off and on so fast that the Whedcos don't respond correctly. A small capacitor across the outputs of the optoisolator slows down their rise time and solves the problem.

4.3.4 Joystick

The Flexbot was designed specifically for experiments in the control of vibration. We are particularly interested in teleoperated control of the robot; that is, a person

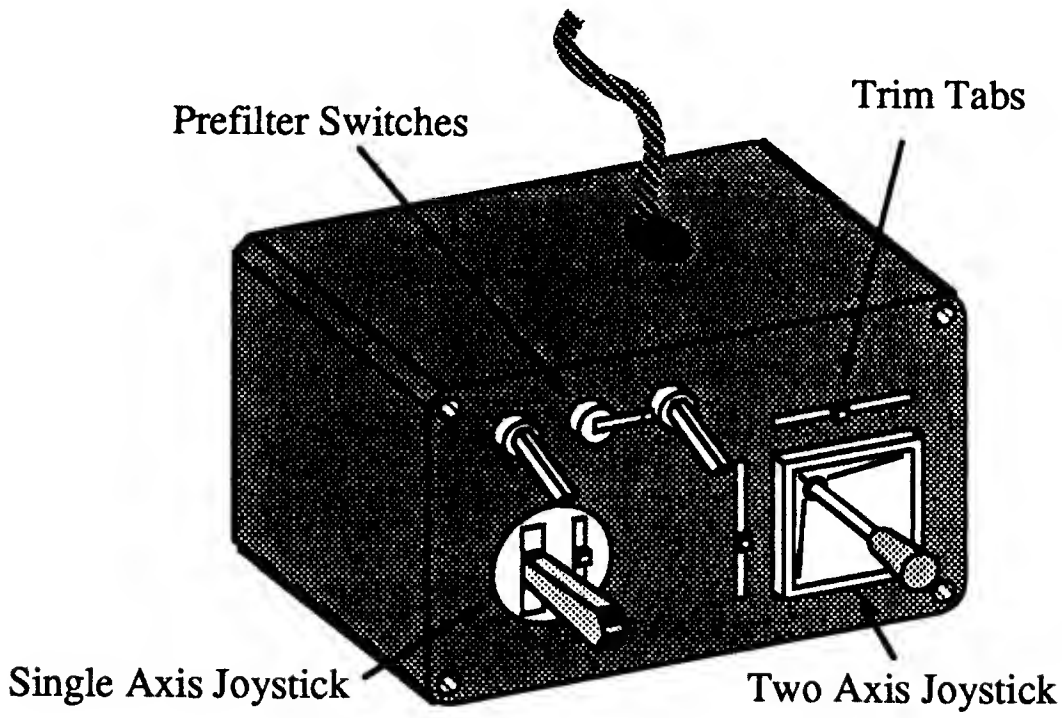


Figure 4.6: Teleoperator control box. Only 6 inches wide, it is designed to be held in both hands and controlled with the operator's thumbs

drives the robot around and the computer attempts to control the vibration. For experiments in teleoperation, we built a small joystick box, shown in Figure 4.6. It is designed to be held with both hands (6.25 in x 3.75 in x 2.5 in). The operator uses the 2 axis stick in the right hand to control cartesian motion of the end of the robot in the plane perpendicular to gravity and the single axis stick in the left hand to control the up and down motion. Section 4.5 will go into more detail on how the joystick is used to control the robot.

The joysticks are spring-return potentiometers sticks manufactured by Kraft Systems. Their range of motion is 27 degrees and they each connect to a 5 kΩ pot. The two axis stick is a Model 910211–13, which has a trim tab and a square mounting bezel. The single axis stick is a Model 910111–13. The joystick box also has three small switches. Each switch is connected to a pull-up resistor and returns a signal appropriate for the MVME 340A digital I/O board. The joystick box plugs into an interface box in the circuit box section of the rack which supplies them with 5 volts of DC power. The interface box routes the signals from the joystick to the

correct VMEbus cards. The potentiometer signals from each axis are fed into A/D converters. The switch signals are fed to the MPP board.

4.3.5 Microswitches and the Protection Circuit.

There are a few other circuits worth mentioning. There are microswitches in the base of the robot which warn the computer that the robot is nearing an emergency stop, and there is a protection circuit to stop the robot if the computer crashes.

The base and first joint each have a pair of microswitches that are contacted just before the robot reaches an emergency stop. They were originally intended to be used for the initial positioning of the robot. The idea is that the robot moves a joint until one switch closes and then reverses and goes all the way around to the other switch. This gives a repeatable test for the position of the robot arm. The circuit for the microswitches is fairly elaborate; each microswitch has two digital lines associated with it. They are powered off of the 5 volt DC supply in the robot base, so they send standard logic level signals to the computer. The first digital line for a microswitch tells the computer whether the switch is open or not. The second tells the computer whether or not the switch is plugged in. This combination allows the computer to read the switches and decide whether or not it should trust the data it is receiving (just in case the operator left a switch unplugged). However, in the current incarnation a set of optoisolators needs to be added at the computer side to guarantee that the PWM amplifier noise does not get connected to the computer ground. As our testing can be done without these circuits, we have left them unconnected.

The protection circuit came from our concern that the computer might crash with a signal on one of the D/A converters which would leave the amplifiers running. The solution was a small box inside of the computer rack that takes as input a signal from the MPP board and has several relay connections as output. Inside of the box is a monostable multivibrator IC running into a buffer that drives a pair of relays. The

monostable is set for retriggerable operation with an output time of 10 milliseconds. Hence an oscillating signal of at least 100 Hertz from the MPP board will cause the output of the monostable to remain high. This drives the output of the buffer chip which then closes all of the relays. The normally open contacts of the relays are brought out as external phono plugs. Any external circuitry that needs to be shut off in case of a computer failure is run through one of these phono plugs.

Originally we had the stop loop from the front panel running through one of the relay contacts. The servo loop supplied the pulse train to the monostable, so the robot would keep the circuit high as long as it was servoing. Unfortunately, the system did not work perfectly. The first problem was that it required the robot to be servoing the joints *before* the amplifiers were turned on. That meant that the robot was liable to take off the instant the start button was pushed. The second problem was that it takes a few tenths of a second for the relay inside of the protection box to cut the stop line. The noise generated by the stop line seemed to cause problems with the CMOS circuitry inside of the protection box, so in some strange circumstances the box would not hold the line properly on or off. In fact, our worst robot crash ever came when I was testing this box and trying to figure out what wasn't working. Since the software that servos the robot has been proven to be reliable, I eventually decided that the box was causing more problems than it solved. At this time we are running without the protection circuit.

4.4 Control Logic

This section deals with the logic used to control the position of the robot. It has three subsections: the servo control loop used to position the arm, an explanation of the cartesian space/joint space positioning of the arm, and some comments on the types of trajectories we generate for the arm to follow.

4.4.1 Servo Loop

The servo loop runs on a single processor board, as described in secton 4.5. For these initial experiments with the robot, we have elected to use a simple PD position servo in joint space with an observer estimating the velocity of each joint. We chose the PD servo loop because it is easy to implement, well understood, commonly used in robotics, and does not require a great deal of computation. The servo loops were designed assuming a rigid output; no knowledge of the dynamics of the system was taken into account.

Original PD loop design on the Flexbot was done by Erik Vaaler. This control loop still runs on the base joint and first joint. The same basic structure was modified to run on the elbow joint. The amplifiers are current amps, hence the plant of the control loop can be modeled as a double integrator. The servo loop runs at 1000 hertz, essentially as fast as we could make it. It uses floating point arithmetic because our processor boards have floating point co-processors and we do not suffer a time penalty for working in floating point. Design and initial experimentation of the servo controller was done on the MatrixX program. The regulator gains were specified by a poleplace method and the observer gains by an LQG design that takes into account the quantization noise from the encoders.

The observer reconstructs the velocity of the system. It is a standard, full order observer with poles at $0.0429 \pm 0.1417i$ for the two base axes and $0.0230 \pm 0.1055i$ for the elbow joint; the poles are in the Z-plane. The regulator poles for the first two axes are at (0.7, 0.8) and for the elbow joint are at (0.95, 0.98) in the Z-plane. The elbow joint poles ended up being much farther out because of an instability associated with the second mode of vibration of the second link of the robot. Because we do not take into account the dynamic behavior of the robot in our controller, we had to lower the gains.

In practical terms, the gains of the robot are tabulated in Table 4.1. An important

The base joints	Position stiffness	5038	ft-lb/rad
	Velocity damping	103	ft-lb/(rad/s)
	Maximum torque	2.72	degrees
The elbow joint	Position stiffness	200	ft-lb/rad
	Velocity damping	13.9	ft-lb/(rad/s)
	Maximum torque	22.9	degrees

Table 4.1: Robot gains as felt at the output of the joint.

goal for the future would be to improve the stiffness of the elbow joint by modeling the vibration mode and designing a controller that does not excite it.

4.4.2 Joint and Cartesian space positioning

User control of the robot is broken into two standard methods: joint space control and cartesian space control. Joint space control is done in terms of the absolute position of each axis of the robot in degrees. The first and second joints are considered to be at an angle of 0 degrees when they are pointing straight up. Figure 4.7 shows how the joint angles are defined.

Joint space control has software limits built in that keep the robot from running into the base or moving a joint far enough to close an emergency stop. A great deal of the testing of the prefiltering techniques (which will be described in Chapter 5) is done in joint space because it avoids the sticky issues of the filtering of motion near singularities and making sure that the arm always remains in a legal portion of the workspace.

Cartesian space is more restrictive, but better for teleoperation. We use the position of the payload as the definition of where the arm is in cartesian space. Since this is a three degree of freedom arm, the position of the payload at the end of the robot determines the positions of the joints. The position is measured (in inches) in terms of the distance of the payload from the intersection of the centers of rotation

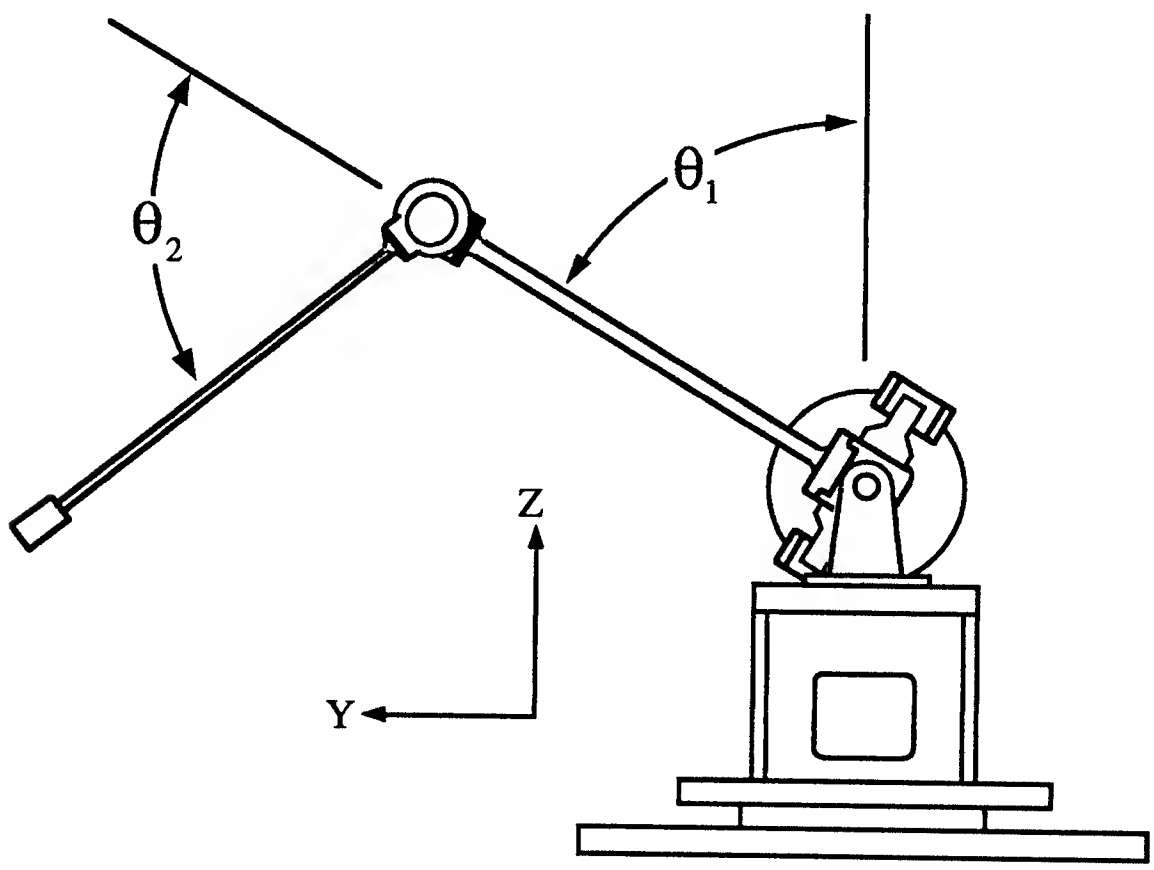


Figure 4.7: Joint positions for the Flexbot

of the base and first joint, as shown in Figure 4.8. The definition of the coordinate frame has the Y-axis pointing towards the operator, the Z-axis pointing straight up and the X-axis pointing to the operator's left.

Software limits keep the arm from getting too close to the robot base or from crashing into itself. Figure 4.8 shows the software limits of the cartesian space of the robot. Besides keeping the end of the robot within this envelope (which is a revolution about the Z-axis) we also constrain the Y position of the robot to be positive. This serves two purposes: First, it guarantees that the inverse kinematics give a unique and consistent solution for the joint angles. Second, the cartesian limits are considered to be "frictionless". That is, if you command the robot to move from one valid location to another valid location but the path passes through an invalid location, the robot will hit the surface and slide along the surface until it can once again travel in the desired direction. Limiting the Y position to be positive

guarantees that the robot will not wedge itself into a corner of the limit surface.

The desired endpoint position of the arm is given as x, y, z in inches. The lengths of the robot links are l_1 and l_2, measured in inches. $\Theta_0, \Theta_1, \Theta_2$ are the required joint angles. The equations for the inverse kinematics of the arm are as follows:

$$\rho = \sqrt{x^2 + y^2 + z^2} \tag{4.1}$$

$$\Theta_0 = \arctan\left(\frac{x}{y}\right) \tag{4.2}$$

$$\Theta_1 = \arccos\left(\frac{z}{\rho}\right) - \arccos\left(\frac{\rho^2 + l_1^2 - l_2^2}{2l_1\rho}\right) \tag{4.3}$$

$$\Theta_2 = \arccos\left(\frac{\rho^2 - l_1^2 - l_2^2}{2l_1 l_2}\right) \tag{4.4}$$

The inverse trigonometric functions are defined so that the inverse tangent returns an angle between −90 and +90 degrees and the inverse cosine returns an angle between 0 and 180 degrees. This guarantees a unique and consistent set of joint angles as long as the endpoint remains within the limit space described in the previous paragraph.

4.4.3 Trajectories

There are two standard types of trajectories that we feed the robot. The first is called a "velocity" trajectory and the second is called an "acceleration" trajectory. The velocity trajectory is specified in terms of a desired position and a velocity. In joint space this velocity is expressed in degrees per second. In cartesian space the velocity is expressed in inches per second. The computer generates a series of set points equally spaced between the starting point and the ending point so that a robot tracking the points exactly would be traveling at the specified velocity. However, since it is a PD servo loop attempting to follow this series of setpoints, the robot will lag behind.

The acceleration trajectory breaks the move into three pieces. It is specified in terms of a desired position, a maximum velocity and an acceleration. It generates a

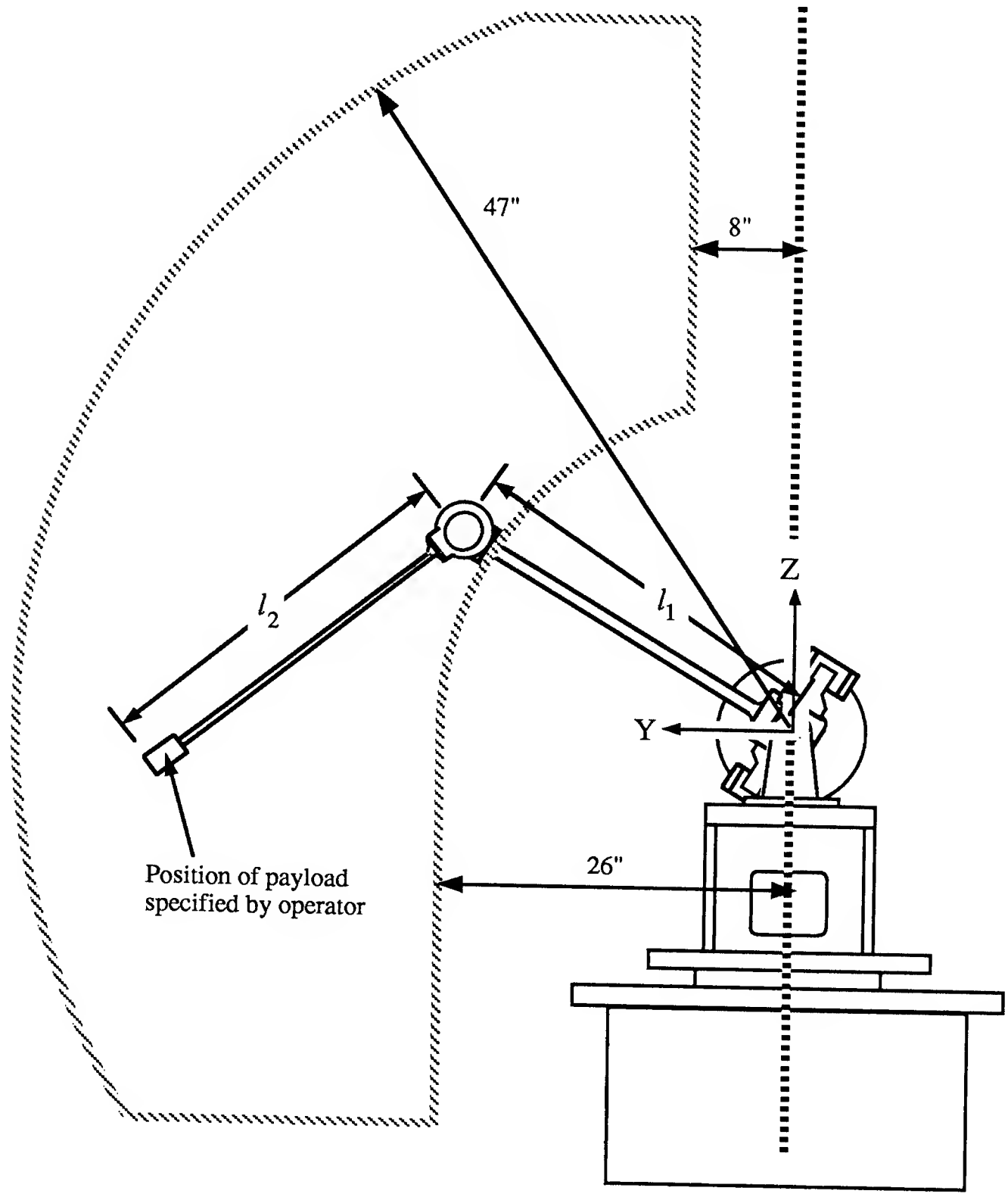


Figure 4.8: Software limits on the cartesian control of the Flexbot. Position is measured at the middle of the payload. The dashed line shows a cross-section of the workspace, which is a section of revolution about the Z-axis.

series of set points like the velocity command, but they are not equally spaced. The trajectory uses the desired acceleration to ramp up to the maximum velocity during the first part of the move, cruises at the maximum velocity for the second part of the move, and then decelerates to finish the move and end up at the desired point. If the acceleration is not sufficient for it to reach the maximum velocity it accelerates for the first half of the move and decelerates for the second half.

In joint space all three robot axes are constrained to reach their destination at the same time; hence only one axis travels at the velocity limit and the other two travel somewhat slower. In cartesian space it is the absolute speed of the end of the robot which is limited; hence typically all axes are traveling slower than the specified limit. In normal operation no provisions are made for acceleration times or for feedback that would insure that the robot is indeed on the desired path. It is a completely open loop process—set points are specified along the desired path and the robot servos as best as it can to the required point.

4.5 Computer Software

This section deals with the computer software that runs the robot. It includes a general background on the operating environment being used and a description of the program that runs on each processor board. All code is written in the C programming language on the Sun under the Unix operating system.

4.5.1 The Condor System

"Condor" refers to the computational architecture and programming environment developed at the MIT Artificial Intelligence Laboratory by Sundar Narasimhan and David Siegel (see [16,17,18]). The Condor system is composed of two parts. The first part is a collection of subroutines and libraries that handle all communication

between programs and the I/O boards. These libraries also provide standard methods of inter-processor communication and timed interrupt routines for servo loops. The second part of Condor is a user interface between the Sun computer and the Ironics processor boards. The interface, called "Xcondor", runs under the X11 windowing system. It opens a window which is connected to a process on the Sun system and windows which are pseudo-terminals that connect to each of the processor boards on the VMEbus system. It also provides fast downloading over the extended VMEbus. The user writes separate programs for each Ironics processor, compiles them on the Sun, runs the Xcondor program, downloads the programs over the VMEbus to the processor boards, and starts them running. He communicates with each program via the pseudo-terminal interface.

Inter-processor communication is a bit of a sticky issue with any parallel computer system, and the problem gets worse when you are trying to use the processors to deal with a real time system servoing at 1000 Hz. There are two standard ways of handling this problem in the Condor system: mailboxing and direct memory access. Mailboxes are part of the hardware of the Ironics processors. A section of their memory has the special property that writing to it generates an interrupt. Mailboxing works by having one processor board write a few bytes of data to the mailbox space on another processor board. Processor #2 is interrupted by the write action, and calls the appropriate mailbox handler. This handler routine does whatever is appropriate for that particular mailbox call and has the option of returning information to the first processor board. Perhaps the most common mailbox routine is to have processor #1 asking #2 where in memory it stores a certain variable.

Mailboxing is great for transferring small amounts of data or signaling that some action should be performed; for example, starting a servo loop or asking where the robot currently is. It is inadequate for fast operation or the transfer of large amounts of data. Now the memory of each processor board is dual-ported; that is,

can be read or written to by any other board in the system. The second technique for interprocessor communication starts with a mailbox routine that runs when the boards first start up. The routine simple asks where in memory a processor is storing a variable. Then during servo loop operation, instead of sending a message via the mailbox to find out the value of the variable, the processor reads the value from memory directly. This can be either a uni- or bidirectional method. Either the memory location can be simply read, or it can be read and then set to zero to provide a simple handshaking mechanism. This style of inter-processor communication is very fast and hence is used when the robot is servoing.

To conclude, the Condor system is a development environment that insulates the programmer from many of the tedious programming tasks: figuring out how to get each board in your system to behave properly, deciding a method of inter-processor communication, writing I/O routines so that the processors in the VME box can talk to each other and the I/O boards, and coming up with timing routines that execute servo loops at the desired frequencies.

4.5.2 The Flexbot Program

The control program for the Flexbot is implemented under Condor. It runs on three Ironics processor boards. The program logic is broken up as follows: The first board, called "Main", handles I/O with the operator, tells the other boards what they should be doing, and generates the trajectories for the robot to follow. The second board, called "Traj", runs at 200 Hz and takes the trajectories generated by Main (which can be in either cartesian or joint space), converts them to joint space if necessary, applies prefiltering techniques to reduce vibration (described in Chapter 5) and stores data on everything that is happening on all three boards. The third board, called "Servo", reads the desired position from the Traj board, the current position of the joints from the Whedco boards, and executes a 1000 Hz PD Servo loop for all three

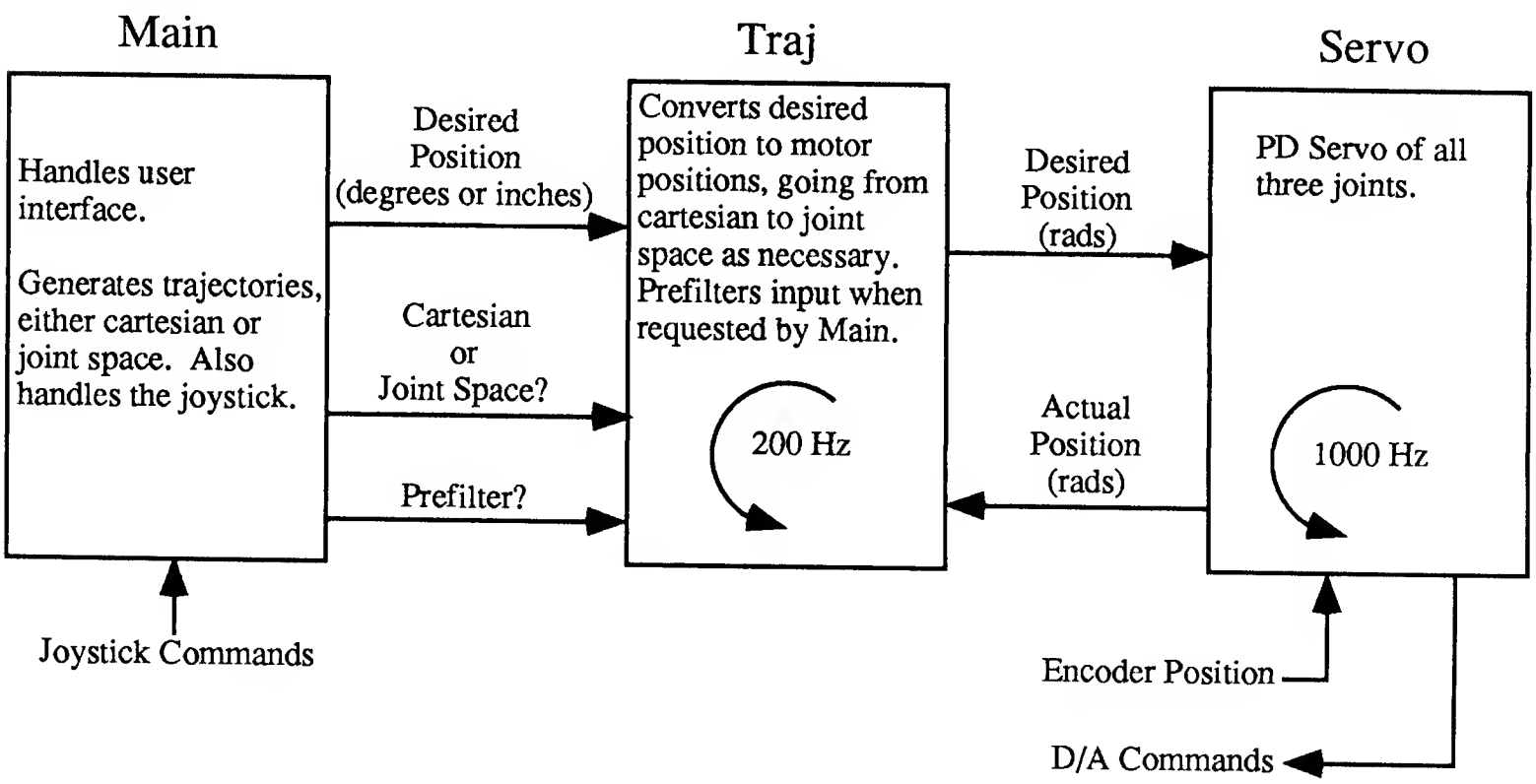


Figure 4.9: Program logic for the Flexbot Controller

joints of the robot. A picture of the process can be seen in Figure 4.9.

4.5.3 Main Board

The Main processor board is where the user controls the system. The Main board controls the operation of the other two boards in the system. Main executes first and when it has initialized it tells the Traj and Servo board to start running. Once both Traj and Servo have reported that they are ready, Main asks the user whether the robot should be run in Cartesian mode or Joint space mode. Then Main tells the other two boards to start their servo loops and prepares to accept user commands.

In joint space, the user can tell the robot to execute a velocity-limited servo or an acceleration move to the desired position. As explained in Section 4.4.3, the joint that has the farthest to go is moved at the velocity limit; the other two joint speeds are scaled so that all joints finish the move at the same time.

In cartesian space the user can either use a velocity or acceleration servo to move

the endpoint of the robot, or he can operate the robot under joystick control. The joystick is run as a velocity controller. The joystick is spring loaded, so when it is in the upright position, the robot remains stationary. Pushing the two-axis lever (see Figure 4.6 on page 73) to the left generates a velocity command moving the end of the robot to the operator's left or X-direction. Pushing all the way to the left runs the robot at the velocity limit to the left. Pushing only part way to the left scales the velocity command accordingly. Pushing forward moves the robot away from the operator in the Y-direction. The single-axis stick runs the robot in the Z-direction, or up and down as seen by the operator. I have been taking an informal survey of all the people that play with the robot. Approximately half prefer pushing the stick away from them to move the robot upwards (the "natural" mode) and half prefer pulling the stick towards them to move the robot upwards (the "airplane" mode). I prefer the natural mode, but I left it as a software option so it can be configured to suit the operator.

Velocity commands are not really commands to the servo loop. What actually happens during a velocity move is the Main board enters into a servo loop running at 200 hertz. At each call of the servo loop, the desired position of the robot is incremented by a suitable amount; i.e., in cartesian space by the velocity command in inches per second divided by 200 and in joint space by the velocity command in degrees per second divided by 200. The Servo board just sees the setpoint of the robot changing. As long as the velocity limits are set to reasonable numbers, the robot does a good job of tracking the desired path.

The Main board also serves two other functions: it informs the Traj board whether or not to use a prefilter and it watches the commanded position of the robot and insures that at no time an impossible position command is issued.

4.5.4 Traj Board

The Traj board handles the fancy computations for the robot. That is, it deals with the inverse kinematics and with the prefiltering. Normal operation is to have the Traj board running a servo loop at 200 hertz. At each invocation of the servo loop the Traj board reads from the Main board the desired position of the robot, either in degrees or in inches. If the position is in joint space the Traj board converts it to a motor position in radians at the joint. If it is in cartesian space, the Traj board uses the inverse kinematics described in Equations (4.1) through (4.4) on page 79 to calculate the desired joint positions.

The Traj Board also handles all prefiltering of the input. Chapter 5 goes into detail on the logic behind prefiltering the input and the different types of prefilters that are used. For now, suffice to say that the Traj board keeps a buffer of the last 5 seconds worth of commands to the robot which is used by the prefiltering algorithm. The Traj board also stores a great deal of other information: commanded torques to the motors, current positions of the joints, commanded positions, and whatever else is called for. Up to ten seconds worth of data is stored by Traj and can be dumped to the Sun system for plotting and analysis.

The math floating-point coprocessor makes the "on the fly" calculation of the inverse kinematics and prefiltering possible; it only takes 1.11 milliseconds to execute the servo loop on the Traj board. That means we could potentially increase the Traj servo speed to 900 hertz or so. This higher rate would not serve any useful purpose as the Main board is only running at 200 hertz. Actually, both boards could be sped up but the system dynamics are so slow compared to the servo frequency that there is no real advantage to running at the faster speed.

The Traj board has one more function which can be selectively turned on or off. As explained in Chapter 2, the endpoint of the robot deflects under gravity. This deflection has three sources: link flexibility, joint spring flexibility and servo

flexibility. Now with a normal PD servo loop, this means that the arm will never actually get to where you commanded it; it will sag. This makes comparisons between commanded position and actual position a little confusing. So we introduced a gravity compensation term. The Traj board takes the commanded position of the robot that is going to be sent to the Servo board and modifies it by the following formulas, where Θ_1 is the position of the motor of the first joint in radians and Θ_2 is the position of the motor of the second joint in radians.

$$\Theta_1 = \Theta_1 - 1.1325 \sin \Theta_1 - 0.0933 \sin (\Theta_1 + \Theta_2) \tag{4.5}$$

$$\Theta_2 = \Theta_2 - 1.6623 \sin (\Theta_1 + \Theta_2) \tag{4.6}$$

Basically the equations are taking into account the torque at each joint that comes from gravity acting on the elbow joint and the payload. Values for the constants were derived analytically and then modified slightly by comparing them with experimental data.

4.5.5 Servo Board

The Servo board has the simplest task of all the boards: it runs a servo loop at 1000 hertz. At each invocation of the loop it reads the desired joint positions from the Traj board, the encoder positions from the Whedco encoder boards, calculates observer estimates, and generates appropriate output voltages on the D/A converter. The actual servo loop design is described in Section 4.4.1.

We have the Servo board running at 1000 hertz, which is about as fast as it can go. It does all calculations in floating point logic with the floating point co-processor. Actual time for each invocation of the servo loop takes 0.7 milliseconds. However, there is a certain amount of overhead time involved with waiting for the VMEbus while other processors are using it, so 1000 hertz represents a realistic upper bound on how fast we can run the servo loop.

Experimental Results

Chapter 5

After spending over a year designing, building, wiring and programming a robot, it would be no fun at all if I didn't do at least a few simple experiments with it. This chapter is about the first series of experiments we have done with the Flexbot. Section 5.1 begins by explaining the different techniques used for vibration control. Section 5.2 describes the vibrational behavior of the arm. Sections 5.3 and 5.4 show results for moving the arm in joint space along a specified path at different velocities and with different vibration control schemes. Finally, Section 5.5 has results for the arm moving in endpoint cartesian space.

5.1 Introduction

There are a great number of people working on methods to control the vibrations of flexible systems. Section 1.2 already mentioned a number of experimental groups that are working on this problem. This section quickly summarizes the different approaches used on this problem and then explains in greater detail the approach we have chosen. For simplicity, the only references listed here are ones that were not already cited in Section 1.2.

5.1.1 A Quick Overview of Vibration Control Techniques

There are a number of methods used to control the vibration in a robot. I arbitrarily divide them into two primary classes: those that use information about the current behavior of the robot and those that do not require information about what the arm is doing. The first class requires some form of measurement from the robot and a closed control loop. Normal ways of obtaining feedback from the arm are accelerometers, optical measurements of endpoint position, or strain gauges that measure link deflection. The measurements are either introduced directly into the control loop or used adaptively to modify the controller gains (Feliu [11], Kotnik [14], and Yurkovich [31]).

For preliminary testing of the robot we decided to go with the second class; simple methods of vibration control that do not require the feedback of information from the robot. This can be done in two ways. Inverse dynamics use the desired output of the system and knowledge of the behavior of the system to derive torque inputs that will reproduce the desired output (Bayo [3,4]). Filtering techniques modify the input to the system to remove energy at the frequencies of vibration (Meckl [15]).

Most of these methods require a good model of the system and quite a bit of computation time to generate the trajectory. For preliminary experiments we decided to use a technique that requires a minimal amount of computation and no feedback from the robot. The technique is called "Impulse Prefiltering" and was developed by Singer [24,25]. This method has the primary advantages of being easy to implement and robust to uncertainties of the system. Currently we are using it only as an open loop process, but it can be implemented in a closed loop controller (Tzes [28]).

5.1.2 Impulse Prefiltering: What is it?

Impulse prefiltering is a relatively new development. The authoritative work on the technique can be found in Neil Singer's Ph.D. thesis [25], but I include a short

explanation here. This section is not meant as a general exposition on impulse prefiltering and why (or whether) it works.

The best way to understand impulse prefiltering is to take a couple feet of string and a small weight. Hold the string with the weight suspended below. Quickly move your hand a few inches and watch the weight swing forward and then back. Now stop the weight and try it again. This time when the weight has swung all the way forward move your hand again so that it is over the weight. If you do it right, the weight will now be motionless and your hand will have moved twice the distance it did originally. That is the basic idea behind impulse prefiltering: break a desired motion into two (or more) smaller steps and then time the spacing between the steps so that at the end of the movement there is no vibration remaining in the system.

Dealing with an arbitrary input (rather than a step input) turns out to be easy. First, imagine a simple system (such as a mass and spring) that vibrates. When it is at rest, give the mass an impulse acceleration. One half cycle of vibration later, give it a second impulse. The second impulse cancels out the vibration from the first impulse, but the system is left with a net input of two impulses. Hence the system experiences a net move, but there is no residual vibration. Now this can be extended to arbitrary inputs. If I give the system an arbitrary input for a short period of time and then repeat the input exactly one half cycle of vibration after I began the first input, each of the little impulses that make up the first input are exactly cancelled by a little impulse from the second input because they occur just half a cycle of vibration later. Hence the system will not vibrate after all of the inputs are finished. Mathematically, this is the convolution of the input function with a pair of impulses. Practically, it means that to implement a prefilter on a robot, you take an arbitrary input and feed the robot the input scaled 50% plus the input scaled 50% with a time lag of half a cycle of vibration. Figure 5.1 is an example of how the impulse prefiltering breaks up a command input into two components, one of which is half a

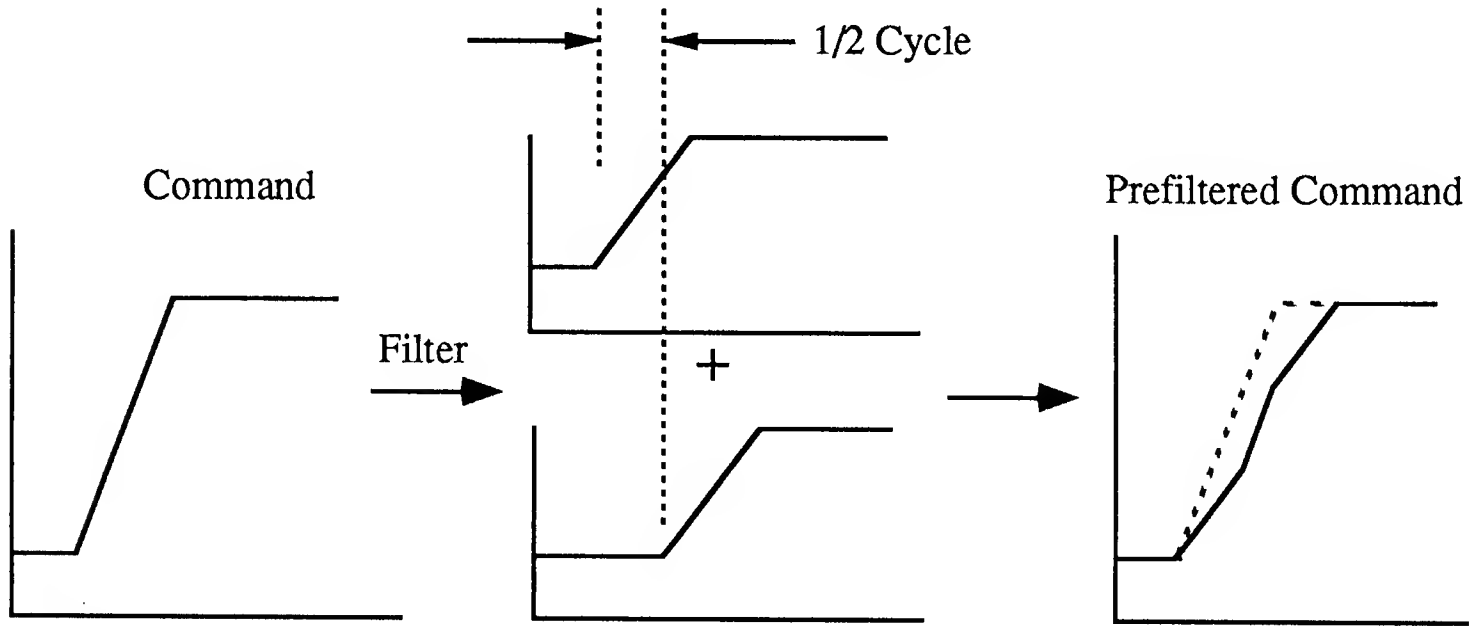


Figure 5.1: An example of how a two impulse prefilter breaks up a command into two pieces and reassembles them into a new command that will not excite as much vibration.

cycle of vibration later than the first. The command that goes to the motor is then the sum of the components. Section 5.2.1 presents some actual data taken from the robot from using this kind of input.

You will note that the system will not have any residual vibration only if three conditions hold: the system must be linear, the vibrational frequency must be known exactly, and there must be no damping. Linearity is necessary so we can justify the superposition of the impulses. The vibrational frequency must be known exactly for all of the vibration to cancel out, or the impulses will not be spaced correctly. Damping is less of a problem: we can still keep the system from vibrating after the motion has been completed by changing the scale of the second impulse. If we know the damping ratio, the height of the second impulse can be made a little shorter so as to exactly cancel the vibration remaining after the first impulse.

It turns out that this simple system will not work very well if the natural frequency of the robot is not known to within 10% or so. But there is a good solution: longer trains of impulses. Instead of using just a second impulse to cancel the first impulse,

we can string together an arbitrarily large number of impulses that give no residual vibration. Most of the filters used in this chapter are of the three impulse variety: that is, three impulses spaced by half a cycle of vibration each. For an undamped system the 3 impulse filter has heights of 0.25, 0.5, and 0.25 respectively, which is essentially a pair of two impulse filters stacked together. An example of such a three impulse filter is shown in Figure 5.2. The three impulse filter has the property of being relatively robust to changes in the natural frequency. Damping modifies the heights of these impulses to look (example: $\zeta = 0.1$) like 0.3344, 0.4877, and 0.1779. Notice that the sum of the impulse heights is always equal to 1. This guarantees that an arbitrary input convolved with the prefilter will always end up at the same spot. For example, convolving a position command to a motor with a trajectory that goes from 0 to 10 degrees will give a trajectory that still goes from 0 to 10 degrees.

A final comment on prefilters: I have explained prefiltering as it applies to a system with a single frequency of vibration. To eliminate residual vibration you convolve your input with a sequence of impulses. For a system with more than one frequency of vibration (such as our robot) there are a number of ways of dealing with the situation. The simplest way to handle two frequencies is to generate separate prefilters for each frequency and convolve them together. The resultant filter will cancel at both frequencies. We do this in Section 5.3 to generate a robust prefilter.

5.2 Vibration Characteristics of the Arm

The vibration behavior of the Flexbot is complicated. One goal of our research was to come up with a map of how the robot vibrates, at what frequencies, and how the vibrations damp out. For initial experiments we set up the Flexbot with the steel links described in Chapter 3, and with stacks of spring washers on the first and second axis. We did not put spring washers in the base axis primarily because we ran out of springs, but partially because the base encoder is used a great deal in our

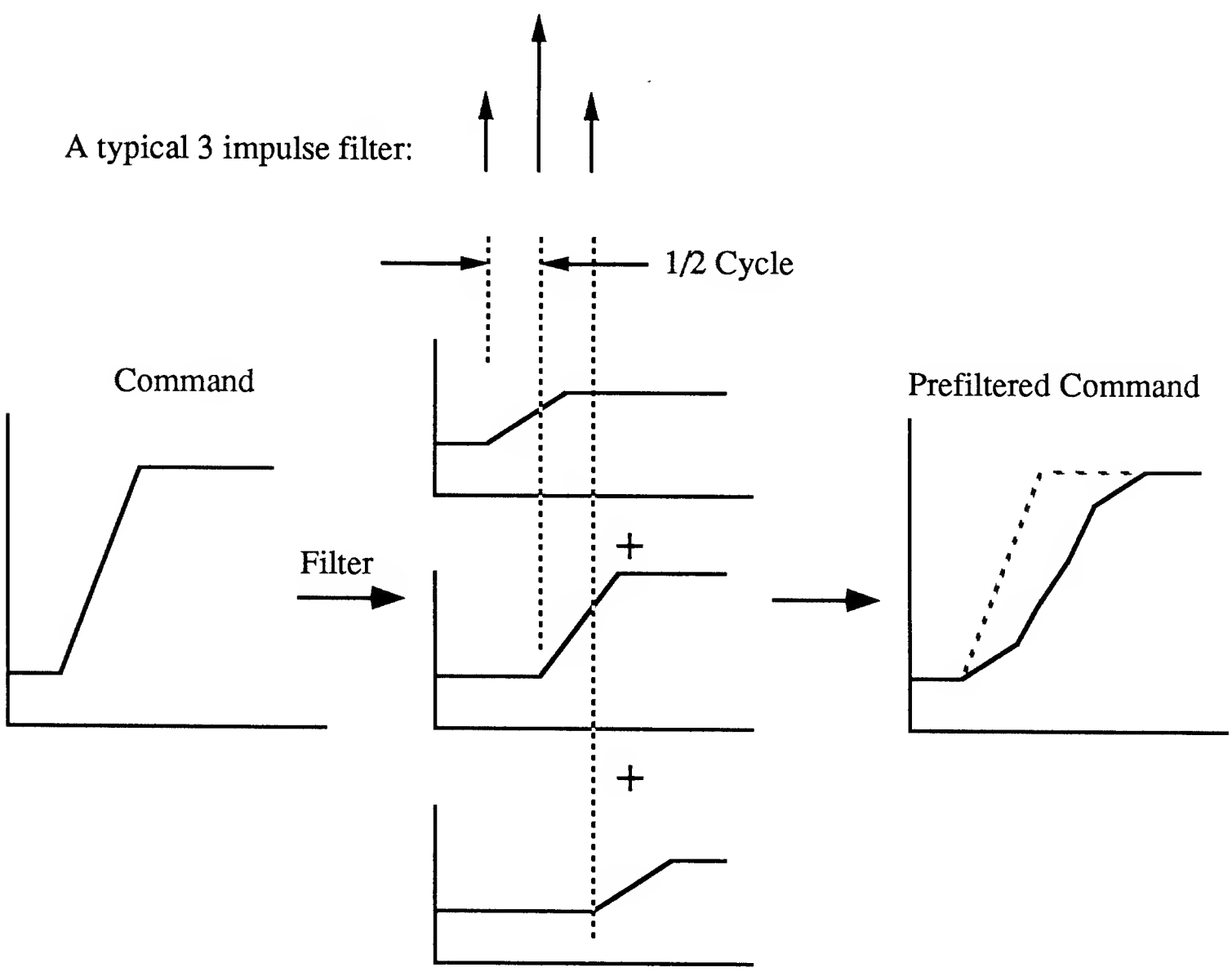


Figure 5.2: An example of how a three impulse prefilter breaks up a command into two pieces and reassembles them into a new command that will not excite as much vibration.

experiments to estimate how the remainder of the robot is vibrating. The springiness of the links is where most of the flexibility of the system is located. The remainder of the flexibility is in the joints, where the PD servo loops contribute more to the flexibility than the spring washer stacks. A description of the servo loops can be found in Section 4.4.1.

To understand the nature of the vibrational frequencies it helps to picture the Flexbot as a system comprised of four masses and three flexible elements. The masses are the inertia of the base motor, the first axis, the elbow joint, and the payload. The flexible elements are the servo stiffness of the base motor, the flexibility of the first axis motor/springs/link, and the flexibility of the second axis motor/springs/link. As you would expect, the lowest frequency of vibration is when all of the masses are moving the same way at the same time. The inertia of the base and stiffness of the base motor's servo is large compared to the other elements in the system, so the second frequency of vibration can be approximated as occuring when the elbow joint and payload are moving in opposite directions.

The links of the Flexbot move in the same plane. This configuration causes the lower modes of vibration to be roughly decoupled from one another. It is convenient to speak of the four lowest modes of vibration as being split into two categories. Keeping the base axis still, picture the plan that links of the robot move in. Two of the directions in which the robot vibrates lie primarily in this plane ("planar" vibrations or "nodding") and two of the directions are perpendicular to this plane ("out-of-plane" vibrations or "wagging"). Technically the modes of vibration cannot be broken up this way, but it is convenient to use this terminology. There are higher frequencies and modes of vibration, but we have found that dealing with just these four is sufficient for the basic control of the robot.

To measure the frequencies of vibration of the arm, we attached an accelerometer to the payload of the robot, positioned it throughout the workspace by 15 degree

increments, and did a frequency analysis of the response of the arm to an impulse. The impulse was supplied either by an impulse torque command to one of the motors or a good hit with a soft hammer. The data was taken with a Brüel and Kjaer Type 4371 Accelerometer connected to a Brüel and Kjaer Type 2651 Charge Amplifier. The resultant signal was fed into a GenRad 2512A Spectrum Analyser, processed, and displayed as a frequency response.

Figure 5.3 is a plot of the natural frequencies obtained from the GenRad Analyser for the planar vibrations. The frequencies of the planar vibrations depend only on the position of the elbow joint, the stiffnesses of the two links, the inertia of the payload, the spring washers in the first and second joints, and the stiffnesses of the PD servo loops in the first and second joints. The position of the two joints at the base has no effect on the vibrational frequencies, so the only variable is the position of the elbow joint. As you recall from Figure 4.7 on page 78, the second link is at a position of 0 degrees when it is parallel to the first link. The vibrational frequencies are symmetric with respect to the elbow position of 0 degrees, so only half of the data is plotted.

When the robot is vibrating out of the plane of motion of the first and second joints, the situation gets more complicated. As the first joint is moved from the home position (0 degrees or straight up) the flexibility due to the servo stiffness in the base changes the frequencies of vibration. Figures 5.4 and 5.5 show contour plots of the experimental data from the GenRad Analyser. The vibrational frequencies are symmetric with respect to the first axis home position of 0 degrees, but they are not symmetric with respect to the elbow joint because for mirror elbow positions the payload mass has a different effective inertia with respect to the base.

As these plots were made from a frequency analysis with an accelerometer, they do not show damping values and in fact are not exactly at the damped natural frequencies observed when you look at data taken from experimental runs. Never-

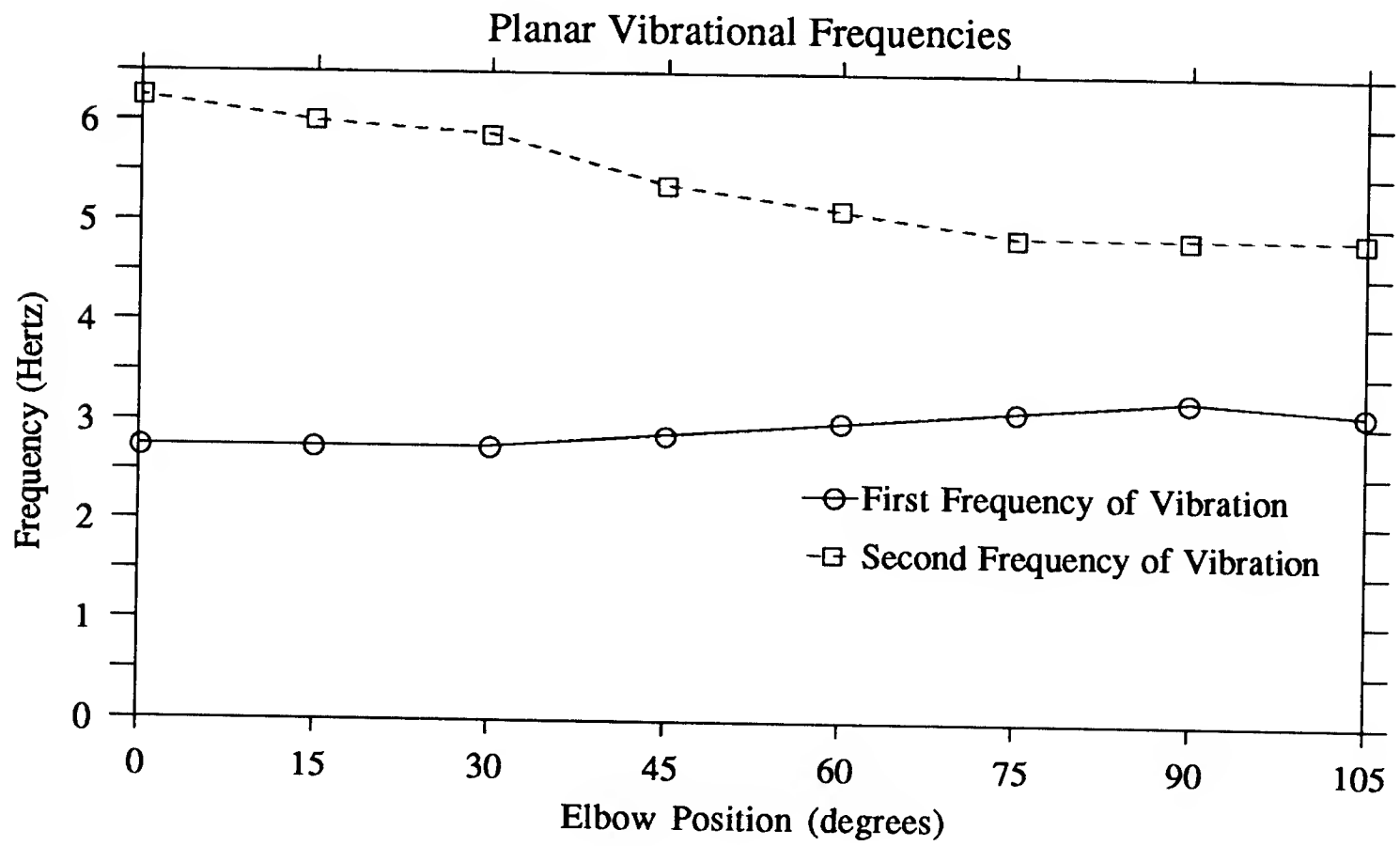


Figure 5.3: Planar vibration data of the Flexbot. The elbow joint is considered to be at 0 degrees when fully extended. The vibration data is symmetric about the 0 degree position of the elbow joint.

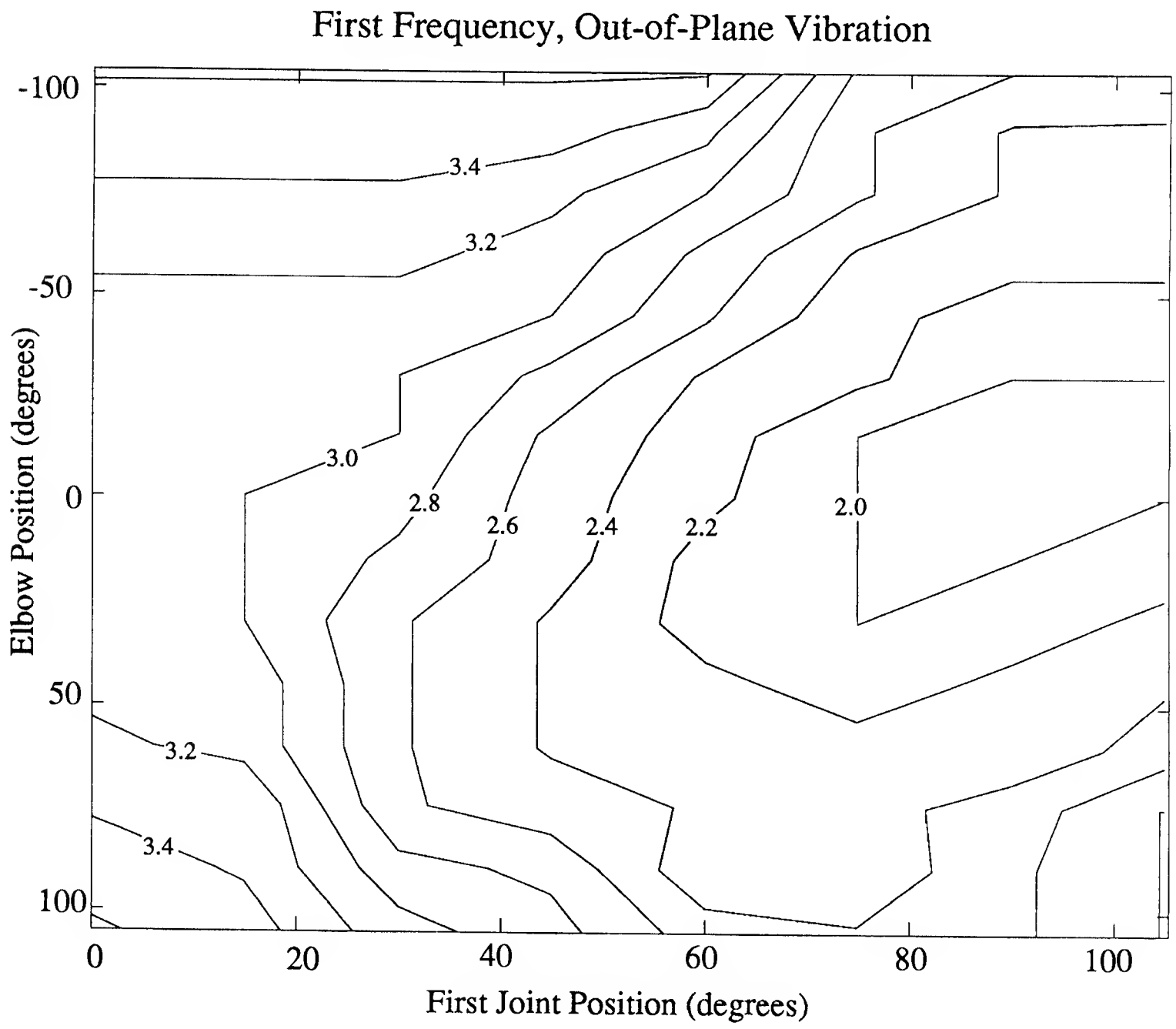


Figure 5.4: Lowest natural frequency of out of plane vibration data of the Flexbot. The first joint and elbow joint are at 0 degrees when standing straight up. The vibration data is symmetric about the 0 degree position of the first joint.

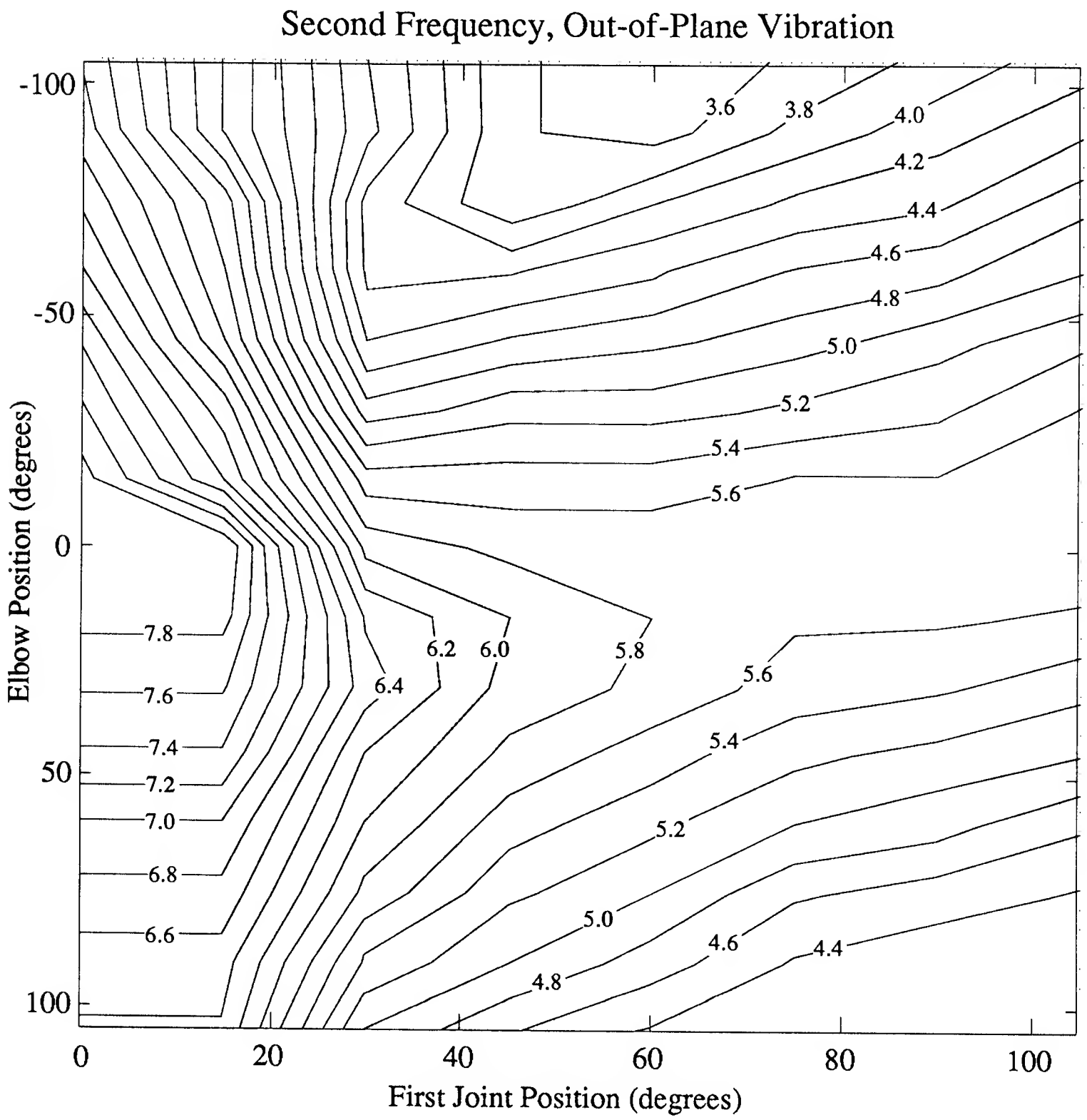


Figure 5.5: Second natural frequency of out of plane vibration data of the Flexbot. The first joint and elbow joint are at 0 degrees when standing straight up. The vibration data is symmetric about the 0 degree position of the first joint.

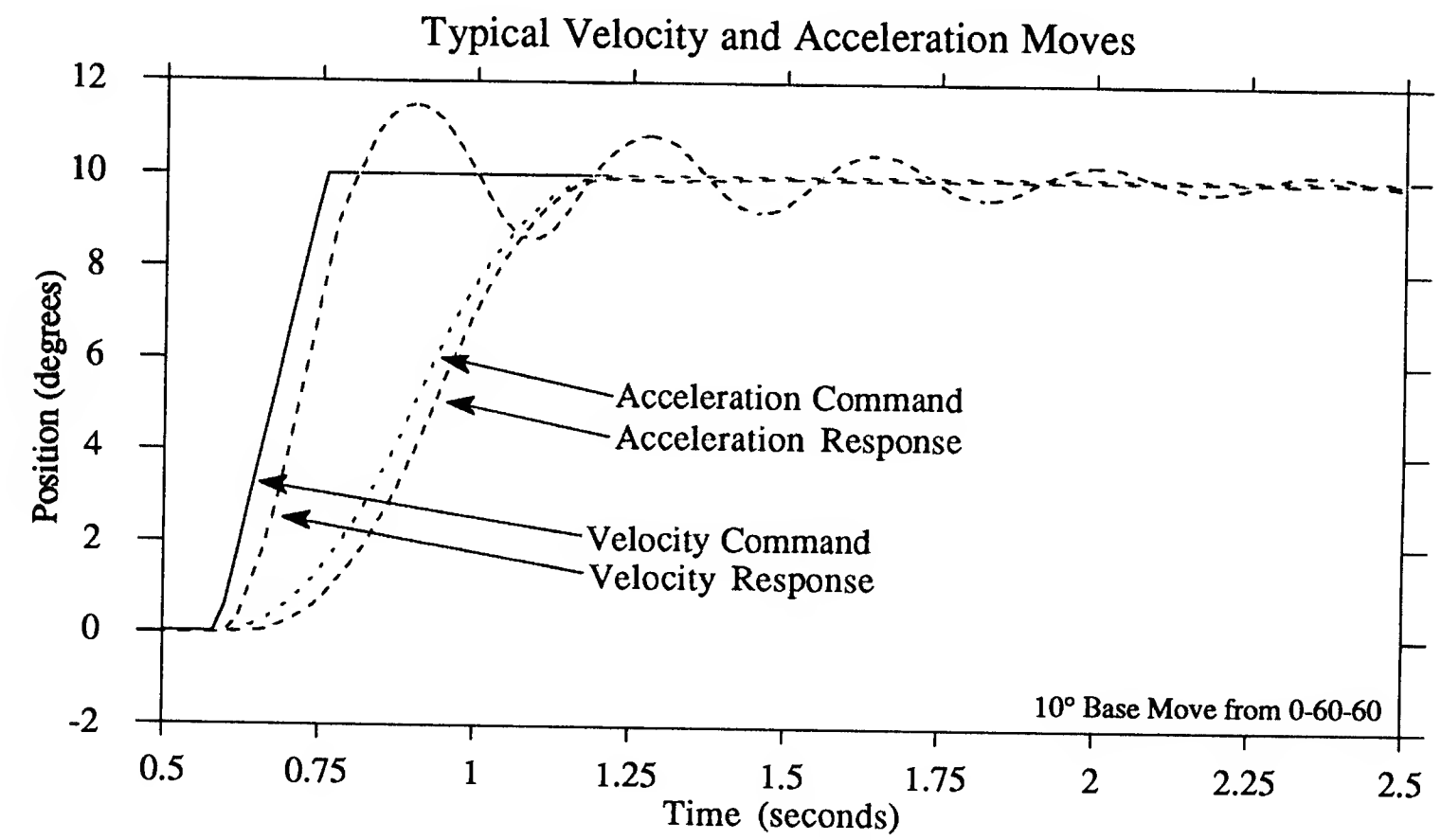
theless, they do give a good idea of how the frequencies change as the arm moves around in its workspace. Damping ranges from a low of $\zeta < 0.01$ with the out of plane vibrations of the arm in the home position to a high of $\zeta > 0.2$ with the arm fully outstretched and vibrating in the plane.

5.2.1 A sample of what move data looks like.

Presenting reams of raw data showing how the robot behaved for different types of moves would be excruciatingly boring after the first dozen or so plots. The later sections of this chapter present move data in a boiled down format which concentrates on how long it took the robot to get where it was told to go and how long it took to stop vibrating. However, you deserve to see at least one piece of authentic robot data, so we present here a typical robot move done in three different ways.

Figure 5.6 shows the three basic types of joint-space moves the robot can perform. The initial configuration of the robot (shown in Figure 5.7) is with both the first and second axes at 60 degrees. The move consists of a command to the base to go from 0 degrees to 10 degrees. The data recorded is the position of the encoder in the base. Data was taken every 0.004 seconds and the encoder has a resolution of 0.009 degrees, so for all practical purposes you may assume that the curves are continuous. For rough comparison purposes, pushing on the payload when the robot is stationary gives a deflection of 5.5 inches per degree of deflection at the base. So the worst move shown in Figure 5.6 generates a payload vibration amplitude of approximately 12 inches peak to peak.

The Velocity command is our baseline move; it shows up on both of the graphs in Figure 5.6. As described in Section 4.4.3, the velocity command consists of a series of equally spaced position setpoints to the robot; in this case a velocity of 60 degrees per second with a 200 hertz servo rate gives setpoints 0.3 degrees apart. The setpoints change quickly enough so that the robot sees an essentially smooth



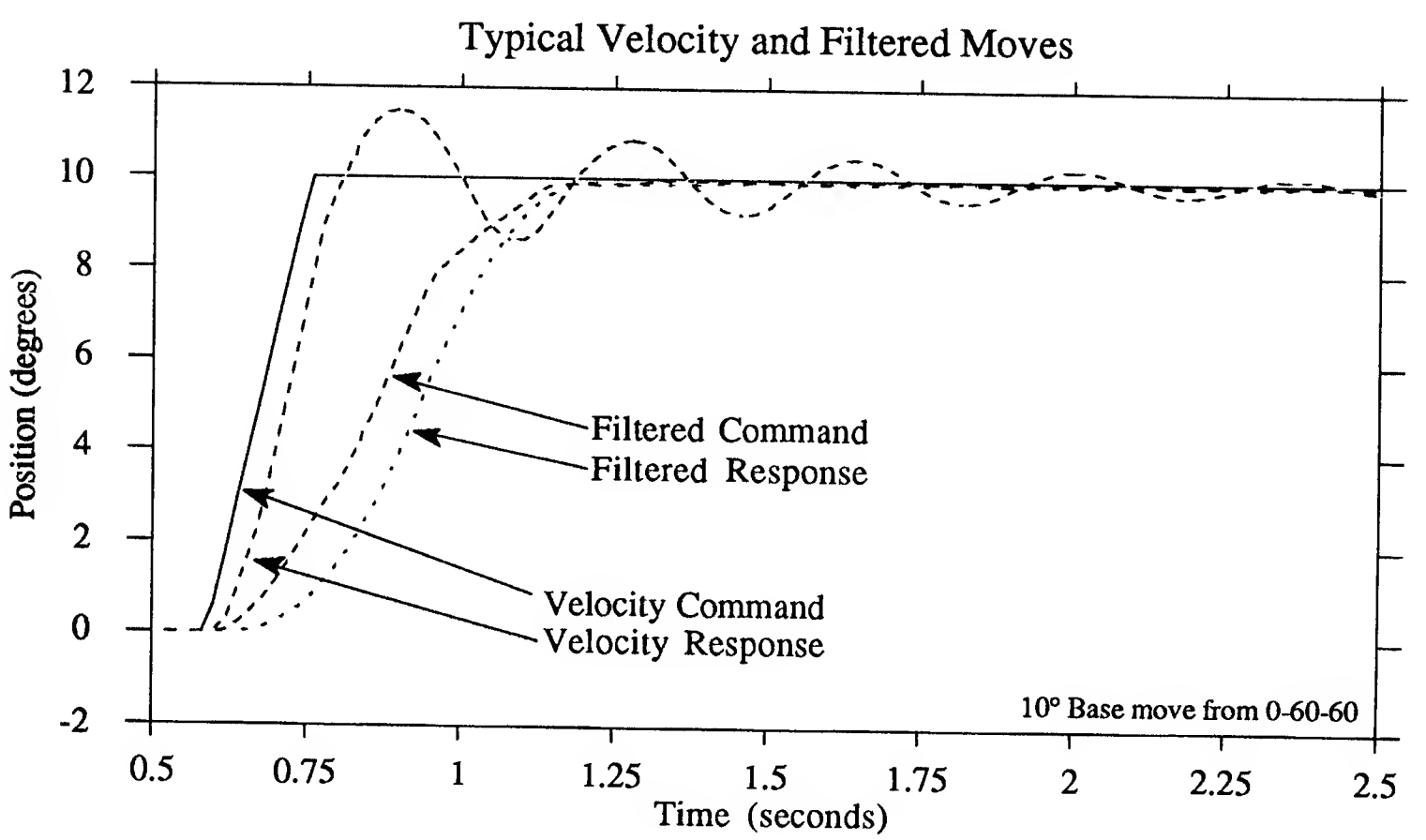
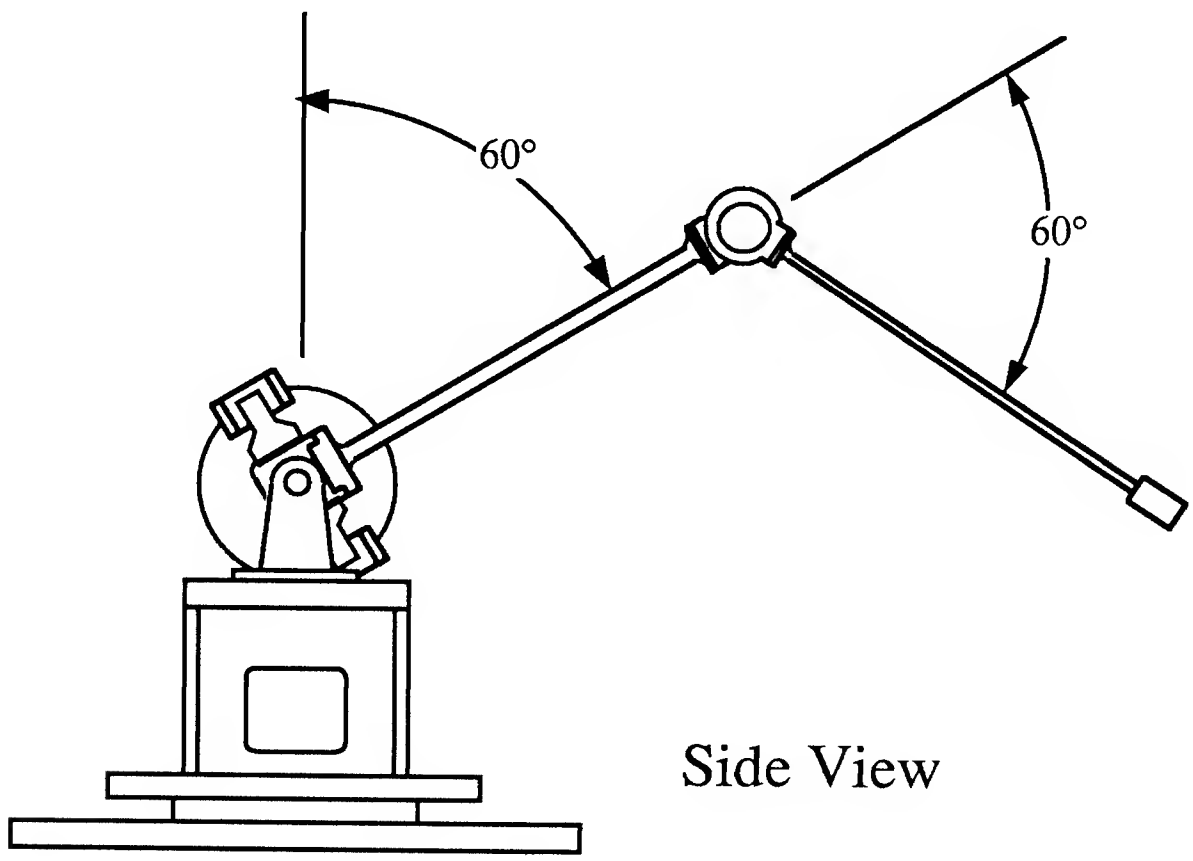


Figure 5.6: An example of vibration data from the Flexbot. The first and second axes are at 60 degrees. The base axis position is plotted versus time.



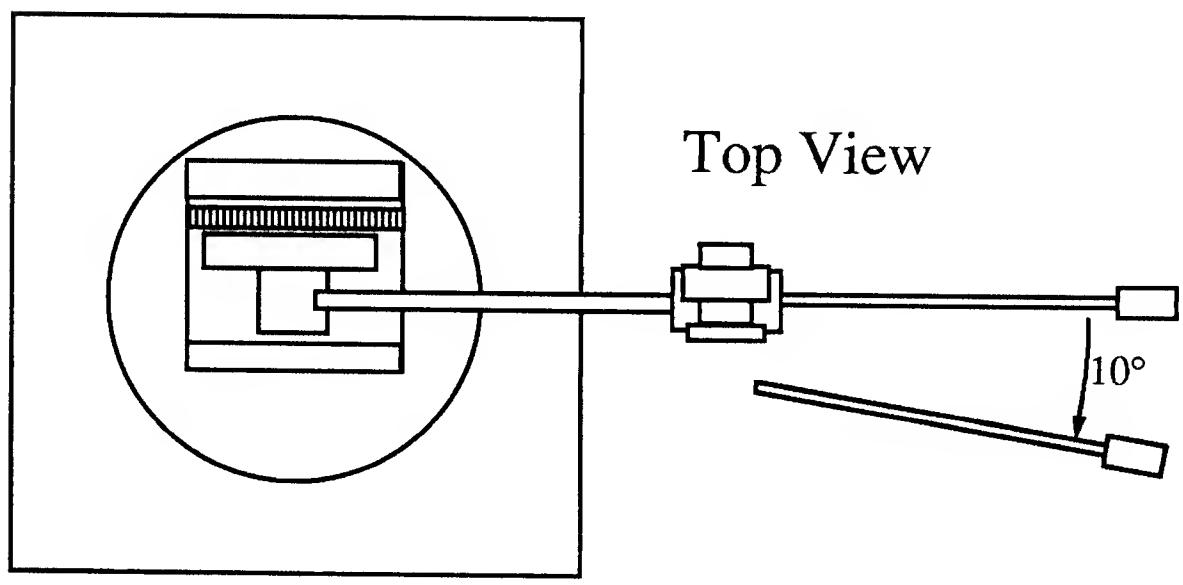


Figure 5.7: Configuration of the Flexbot for the demonstration data. The base axis moves 10 degrees and the other two joints remain fixed.

command. Because the robot is tracking the command with a PD servo loop, it will lag behind the velocity command. As Figure 5.6 shows, the lag time is not relevant compared to the time it takes for the vibrations to settle out. Incidentally, we use a velocity command and not a step command precisely because of the great deal of residual vibration. The arm can be (and has been) bent if you abruptly give it the full torque available from the motors. Our normal testing procedure is to repeat a move many times at a slightly faster velocity each time. Eventually the endpoint vibration starts to look dangerous and we stop—usually around an amplitude of 12 inches peak-to-peak.

The Acceleration command is an improvement on the velocity command because we do not assume that the arm can leap to the desired velocity instantaneously. As described in Section 4.4.3, the acceleration command ramps the velocity up at a constant acceleration to a specified maximum velocity, runs at the maximum velocity for a while, and then ramps the velocity back down to stop the robot at the desired position. Combined with the effects of the PD Servo loop, this results in a smoothly applied torque to each joint. The upper graph of Figure 5.6 shows an acceleration move of 100 deg/sec^2 and a maximum velocity of 40 deg/sec. This results in a very smooth motion compared to the motion of the velocity command. The price of the reduced vibration is the time penalty for accelerating up to speed at the beginning of the move and decelerating at the end.

The Filtered command shown in Figure 5.6 is a convolution of the velocity command with a three impulse prefilter that is tuned to the natural frequency and damping of the system: in this case, 2.75 Hz and 0.10 damping. If you examine the figure closely, you can see how the prefilter breaks the move up into three distinct segments. Although the filtered movement takes 0.364 seconds longer to complete than the unfiltered movement, it has virtually no residual vibration.

Throughout this chapter we will be comparing filtered commands with accel-

eration moves, or even filtered acceleration moves. Filtering works extremely well at cancelling out vibration at the specified frequency, but the extra time required is equal to a cycle or two of vibration. It does not do anything about higher frequencies so they end up being the limiting factor on how fast you can move the arm. Acceleration moves tend to not excite the higher frequencies of vibration, but to avoid exciting the lower frequencies you must accelerate relatively slowly. For example, Figure 5.6 originally had an acceleration of 200 deg/sec^2. That gave a move time shorter than the filtered move, but the level of residual vibration was almost equal to that of the velocity move.

5.3 Constant Vibrational Frequency Move

Our first series of experiments deal with reducing residual vibration in a "long" move where the vibrational frequencies of the arm do not change over the course of the move. A long move is one where the arm ramps up to speed quickly, travels at a maximum velocity for a substantial portion of the move and then ramps down again. We chose this type of motion (as opposed to a short move) because it allows great latitude of choice in different accelerations, lengths of prefilters, and maximum velocities. It is also the rough equivalent of the behavior of many industrial controllers.

All of the data in this section comes from one basic move: swinging the base axis of the robot through 100 degrees. The first and second joints were both set to 60 degrees. The base joint moves from 0 to 100 degrees and stops. Figure 5.8 shows the configuration of the arm. Vibration data is recorded from the optical encoder of the base joint. The data from each move is compiled down into two numbers: the first number is how long the command took, and the second is how long the vibrations took to die down after the command had been finished. That is, the commanded time is the time between when the controller told the arm to start moving and when the controller told the arm to stop. The settling time is measured from when the

controller reaches the stopping point to when the vibrations have damped down to a specified level. The sum of the commanded time and the settling time is referred to as the overall move time.

In this section we measure the settling time by waiting for the amplitude of the vibrations of the base joint to drop to less than 0.10 degrees peak-to-peak. This agrees well with visual estimates of how long the robot took to stop vibrating. It also correlates with the amplitude of vibration found at the joint immediately after the motion has been completed. We did not use a measure of the amplitude of vibration because the time lag inherent in the PD servo makes it difficult to accurately measure the residual vibration just after the commanded move has finished. It also does not take into account higher frequencies of vibration; they may have large amplitudes, but they decay quickly. Since our emphasis is on how long it takes until the robot has stopped vibrating, we find it more appropriate to use the 0.10 degree criteria. We chose 0.10 degrees because it is the smallest amplitude of vibration that can be measured reliably by the encoders (which have a resolution of 0.009 degrees).

Figure 5.9 shows the vibration remaining at the end of two typical 100 degree moves. The upper graph shows the resulting vibration of a 60 deg/sec velocity move. In this case, the commanded time is just 100 degrees divided by 60 deg/sec, or 1.67 seconds. The settling time can be measured at 1.65 seconds which gives an overall move time of 3.32 seconds. Clearly this move has great deal of residual vibration at the 2.75 hertz frequency. The lower graph shows a typical prefiltered move, as described in Section 5.3.2. The velocity command to the prefilter was 0.56 seconds long and the three impulse filter has an overall length of 0.36 seconds. The commanded move time is the sum of these two, or 0.92 seconds. The settling time can be measured as 0.69 seconds which gives an overall move time of 1.61 seconds. The residual vibration is lower for this move and occurs at the 6.0 hertz vibrational frequency.

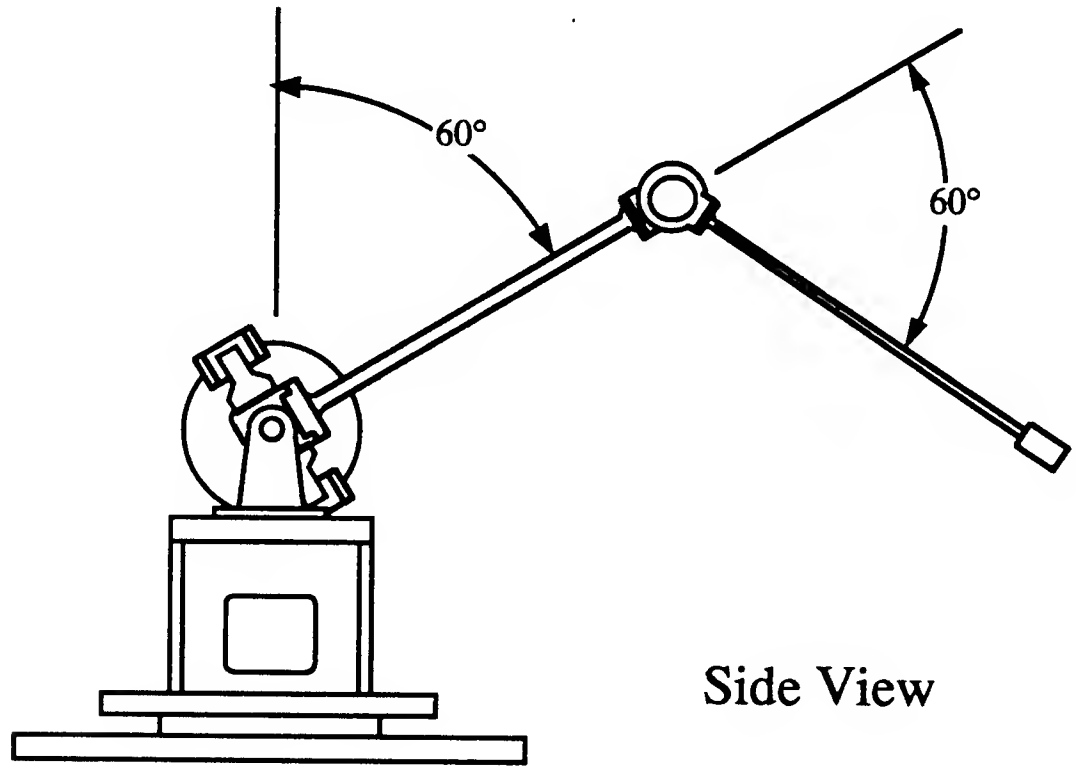


Figure 5.8: Configuration of the Flexbot for the constant vibrational frequency moves. The first joint and elbow joint stay fixed at 60 degrees. The base joint moves 100 degrees.

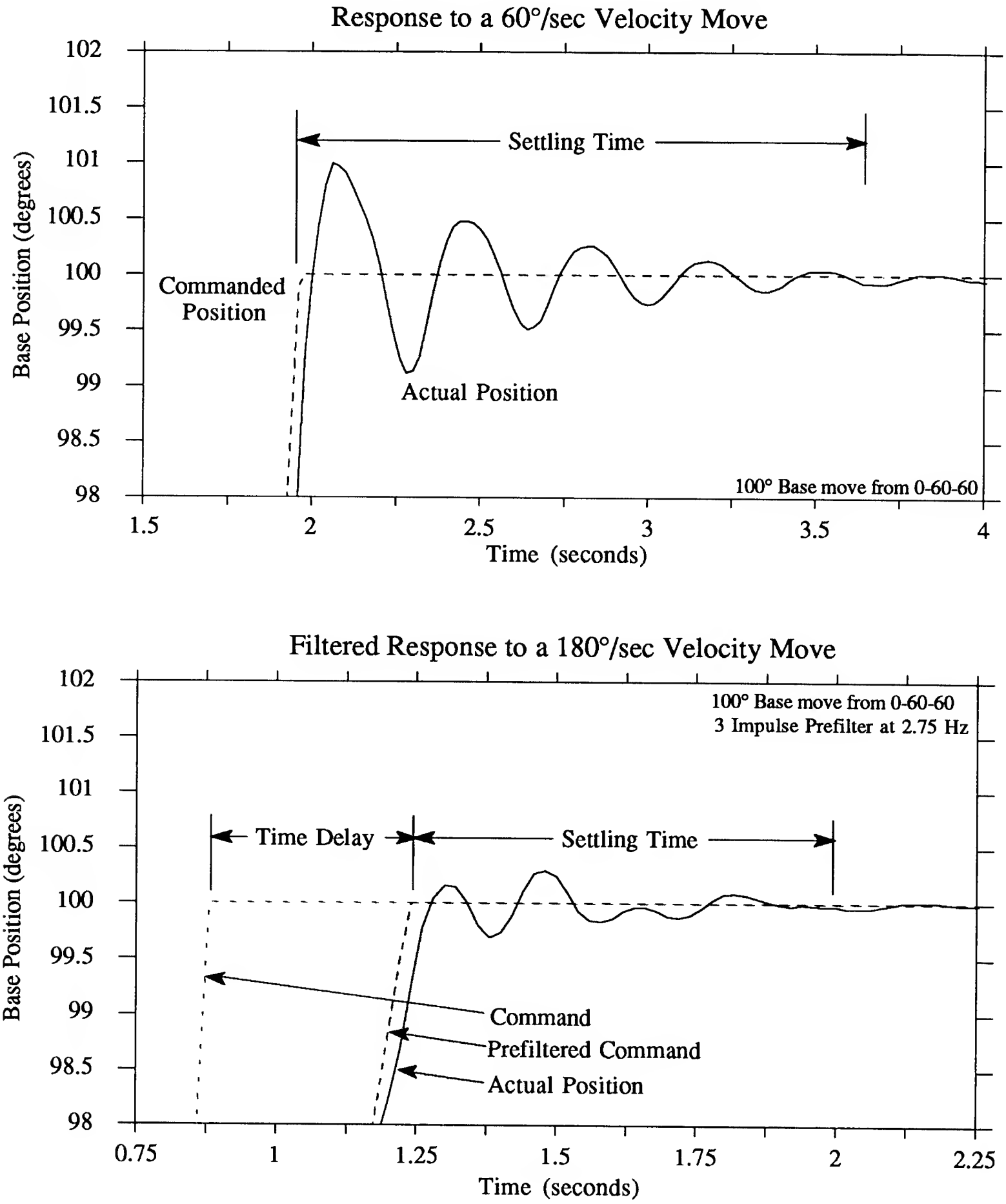


Figure 5.9: Sample responses to 100 degree base moves. The upper graph shows the residual vibration from a 60 deg/sec velocity move. The lower graph shows the residual vibration from a 180 deg/sec velocity move that has been prefiltered by a 3 impulse filter tuned to 2.75 hertz.

5.3.1 Basic move data at different velocities

For the first set of experiments we ran the 100 degree move using velocity commands and acceleration commands. Figure 5.10 shows the compiled results of these tests. The upper graph gives the results for the overall length of time that the move took to finish plotted against the maximum velocity obtained by the robot during the course of the move. The lowest line in the graph is the "Optimum Move Time". You can think of it as the length of time a rigid robot would take to complete the 100 degree move at the given velocity.

The lower graph of Figure 5.10 shows the settling time for each of the types of moves. Notice that the slow acceleration move and velocity move do not extend past 100 deg/sec. The slow acceleration move does not go past a speed of 100 deg/sec because at this speed and acceleration it takes the robot 50 degrees to accelerate up to 100 deg/sec and it then immediately has to start decelerating. The velocity move does not extend past 100 deg/sec because at this speed the amplitude of the tip vibration was almost 15 inches. Going faster would be ill-advised.

It is not suprising that by accelerating the arm smoothly up to speed you do not induce as much residual vibration as giving the arm a velocity command. But it is confusing that the amount of residual vibration induced does not increase monotonically with the increase in move velocity. At first glance, one would think that the higher velocity would induce more vibration because the arm takes longer to accelerate up to speed. Why does the residual vibration go almost to zero at 70 deg/sec? There are two answers to this question, one of which is relevant to all types of moves and one that explains why this particular speed and acceleration combination leaves no residual vibration.

First, let us discuss the phase behavior of the residual vibration. Figure 5.11 shows the commanded and actual positions of the base for a typical 60 deg/sec velocity move. It also shows the difference between the commanded position and the

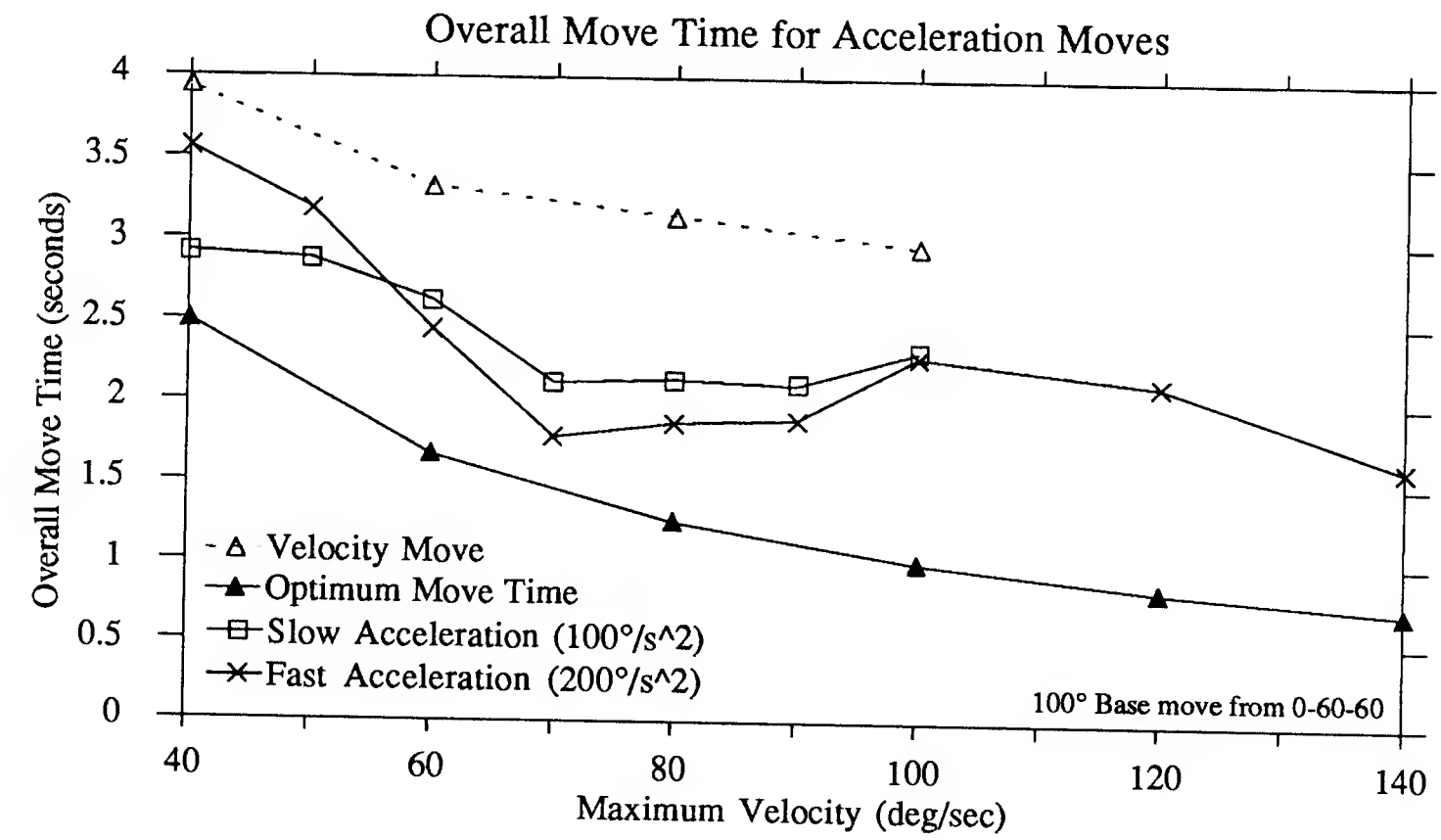


Figure 5.10: General velocity and acceleration moves for a 100 degree move of the base joint with the first and elbow axes at 60 degrees apiece. The upper graph shows the time required for each move from start to the finish of vibration. The lower graph shows how long it took the vibration to settle out.

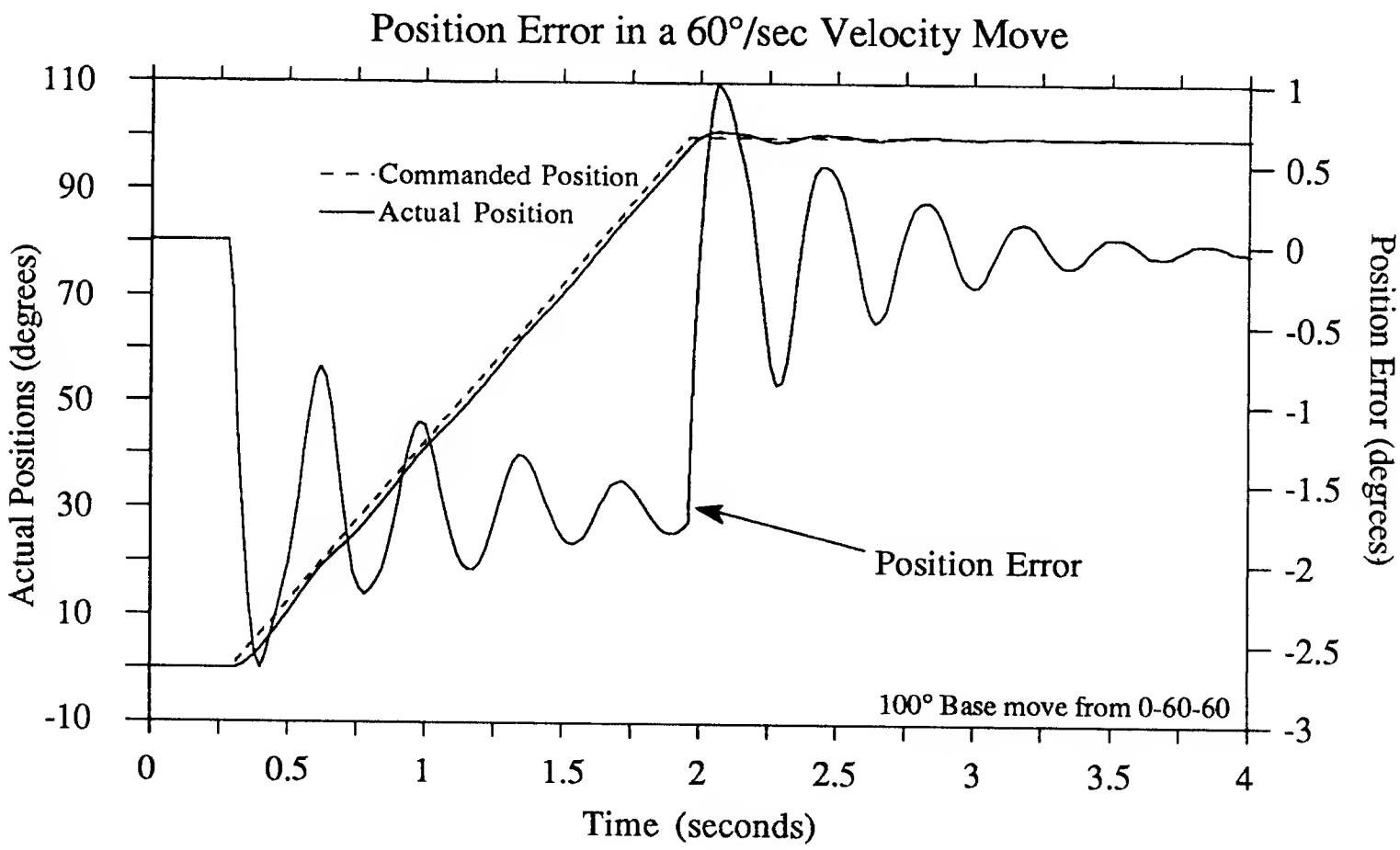


Figure 5.11: Behavior of a 60 deg/sec velocity move of 100 degrees. The position error is the difference between the commanded position and the actual position.

actual position. As you can see, the error in position oscillates about the position of -1.6 degrees during the course of the move and about 0 degrees at the end of the move. The acceleration of the arm up to speed has induced a vibration and then the deceleration induces more vibration. In this particular instance the travel time is long enough so that most of the oscillation from the initial acceleration has had time to damp out before the arm decelerates. But in general the phase of the vibration remaining from the acceleration is important because this vibration will be superimposed upon the vibration that comes from the deceleration.

Figure 5.12 is a comparison of the effects of the phase of the initial acceleration vibration on the settling time. For these experiments we put the arm in the normal position (elbow joint and first joint at 60 degrees) and moved the base joint with a constant velocity move of 50 deg/sec for different move lengths. The acceleration

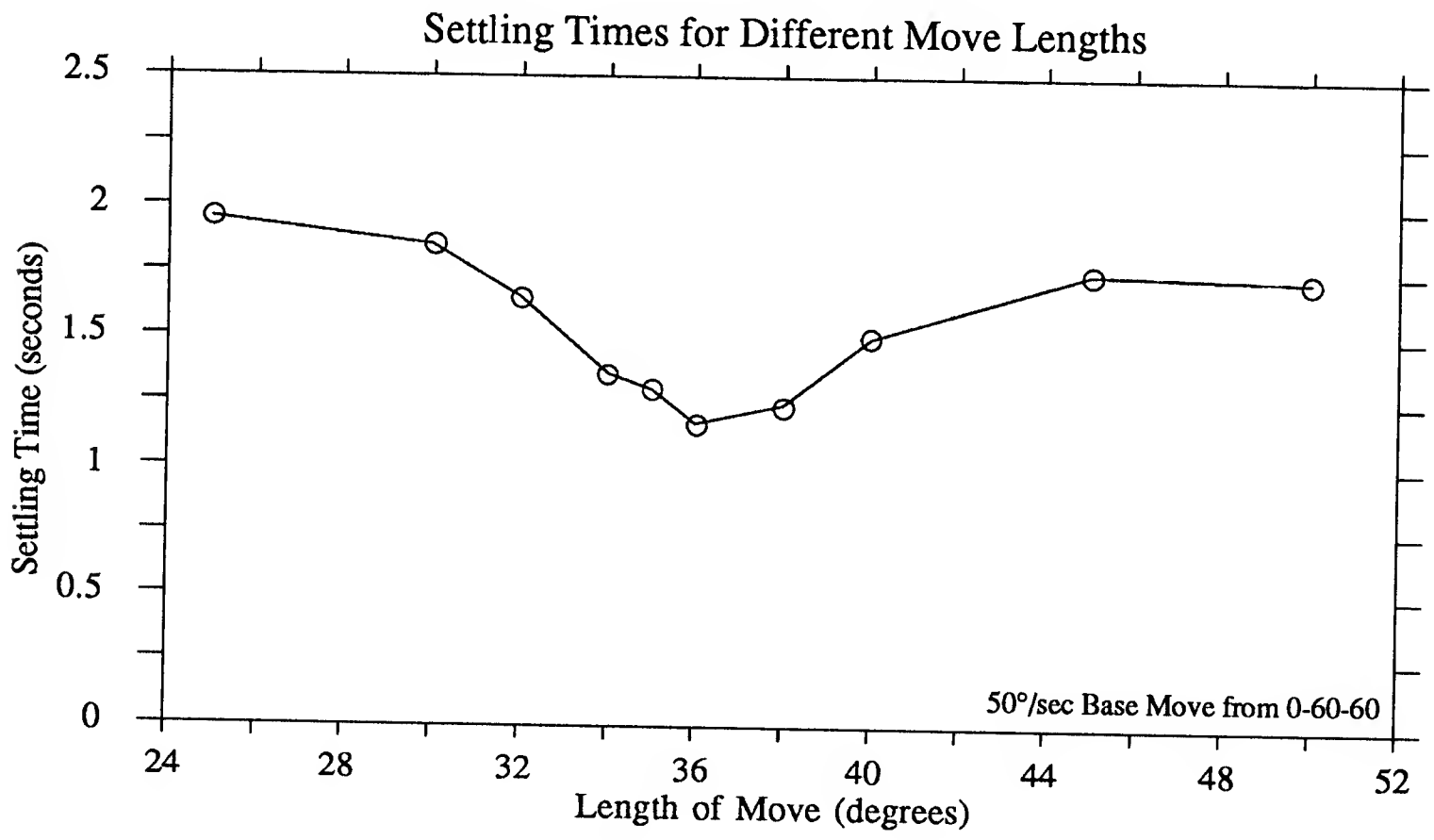


Figure 5.12: A 50 deg/sec velocity move with different move distances. The dip in the settling time corresponds to the residual vibration from the initial acceleration canceling with the residual vibration from the deceleration

and deceleration times are the same for all of these moves; the only difference is in the travel time between acceleration and deceleration. As you can see, there is a pronounced dip in the settling time right at a move distance of 36 degrees. This corresponds to a commanded move time of 0.72 seconds which is just two full cycles of the 2.75 hertz frequency of vibration. The settling time is not 0 at this point because the vibration from the acceleration has been partially damped out. What this graph shows is that the amount of residual vibration is sensitive to the move time.

In the case of the acceleration move of 200 deg/sec^2 and a maximum velocity of 70 deg/sec (shown in Figure 5.10) there is no residual vibration to speak of. Hence it cannot be possible that the vibration remaining from the acceleration is canceling with the vibration from the deceleration because the acceleration vibration will been

partially damped out. Instead, what happens is the vibration cancels itself out during acceleration and hence there is no residual vibration during the constant velocity part of the move. The deceleration is the mirror image of the acceleration, so it too cancels out all residual vibration.

Why does this happen? In effect, you are witnessing the results of a very simple impulse prefilter. At the particular speed of 70 deg/sec, the fast acceleration move takes just 0.35 seconds to accelerate up to speed; this is approximately one full cycle of 2.75 hertz vibration. This can be pictured as a move with an acceleration of 400 deg/sec^2 that lasts for 0.175 seconds but has been prefiltered by a 2 impulse filter with the pulses that are 0.175 seconds apart. Because the prefilter corresponds closely to the actual cycle time of 0.182 seconds for a half cycle of vibration, the net result is a move where a great deal of the vibration has been canceled out.

At 70 deg/sec, the slow acceleration move takes 0.7 seconds to accelerate up to speed. This can be imagined as a move with an acceleration of 400 deg/sec^2 that lasts for 0.175 seconds but has been prefiltered by a 4 impulse filter where the pulses are of equal height and the spacing between each pulse is 0.175 seconds. This is effectively a pair of 2 impulse filters. The first pair cancels vibration and the second pair also cancels vibration. To eliminate more of the residual vibration, the spacing should correspond to half a cycle of vibration or 0.182 seconds. This error in timing means that the filter will not cancel vibration exactly, but the error is compensated for by the slow acceleration rate.

This can be rather confusing if you are not familiar with impulse prefiltering. Let me explain it this way: Picture the fast acceleration (200 deg/sec^2) as actually being two 200 deg/sec^2 accelerations. The first 200 deg/sec^2 acceleration lasts for just 0.175 seconds, which is approximately half a cycle of vibration. The second acceleration lasts for another 0.175 seconds, but the vibration induced by this acceleration is almost 180 degrees out of phase with the vibration from the first acceleration. The

superposition of these two residual vibrations almost cancels out, leaving the system moving at the desired velocity with very little vibration.

Using our knowledge of how the acceleration times can help cancel out residual vibration allows us to specify a criteria for the "best" accelerations. If we specify that the time of the acceleration phase should be an integral number of cycles of vibration, we get the equation

$$\frac{\text{Maximum Velocity}}{\text{Acceleration}} = \frac{n}{\text{Natural Frequency}} \tag{5.1}$$

where n is the number of cycles of vibration during the acceleration phase. This can be rewritten as

$$\text{Acceleration} = \frac{(\text{Maximum Velocity})(\text{Natural Frequency})}{n} \tag{5.2}$$

This equation also depends on whether or not the arm can get up to the desired speed within the move distance. The necessary criteria can be calculated as

$$\text{Maximum Velocity} < \frac{(\text{Move Distance})(\text{Natural Frequency})}{n} \tag{5.3}$$

Using Equation (5.2) and our knowledge of the vibrational frequencies of the system, we can select an acceleration for different move velocities that should result in a minimal amount of vibration. It won't necessarily result in the lowest possible residual vibration for a constant acceleration up to speed because of two factors. First, high acceleration rates induce vibration at the higher frequencies which are not being compensated. Second, the model is not accurate. The real system has damping and inaccuracies associated with picking out the frequency of vibration. These are not taken into account here.

Figure 5.13 shows the result of using Equation (5.2) to pick the acceleration, given that we want to accelerate up to speed within one cycle of vibration, or $n = 1$. This is the "variable acceleration" line on the graph. We call it variable acceleration because there is a unique acceleration for each velocity on the graph. As you can see,

the amount of residual vibration is negligible at speeds below 80 deg/sec. At higher speeds the inaccuracies of the modeling and the generally high accelerations induce a reasonable amount of residual vibration, but still not as bad as the fast acceleration move where the acceleration is constant. At the highest speed of 140 deg/sec, the fast acceleration move is better. This happens because it is approaching the criteria for having an acceleration time of *two* cycles of vibration. We did not plot the variable acceleration curve for $n = 2$ because although they have less residual vibration than the $n = 1$ accelerations, the overall move time is longer by the time of a full cycle of vibration. Besides, there is a better way to do this which we demonstrate in the next section.

5.3.2 Prefiltering the basic move at different velocities

The phase of the vibration at the deceleration time is a factor in how much vibration remains in the system after a move. Perhaps not suprisingly, this is *exactly* the issue that the impulse prefiltering technique deals with. You can think of a velocity move as giving an impulse in acceleration to the system, letting it coast, and then giving a negative impulse to stop it. Then a two impulse prefilter breaks each of the pulses into two pulses spaced so that they cancel out vibration. That means the system comes up to speed with vibration canceled out, runs at the maximum velocity without vibration, and then decelerates from speed with vibration cancelled out. This is just what we were trying to do (albeit imprecisely) with the variable acceleration moves.

Figure 5.14 shows the standard 100 degree velocity move with prefilters that are tuned to remove the 2.75 Hz, 0.1 damping first frequency of vibration. We use two different prefilters: the first is a two impulse prefilter with a total length of 0.182 seconds and the second is a three impulse prefilter with a total length of 0.364 seconds. The upper graph shows the overall time required for the move to be

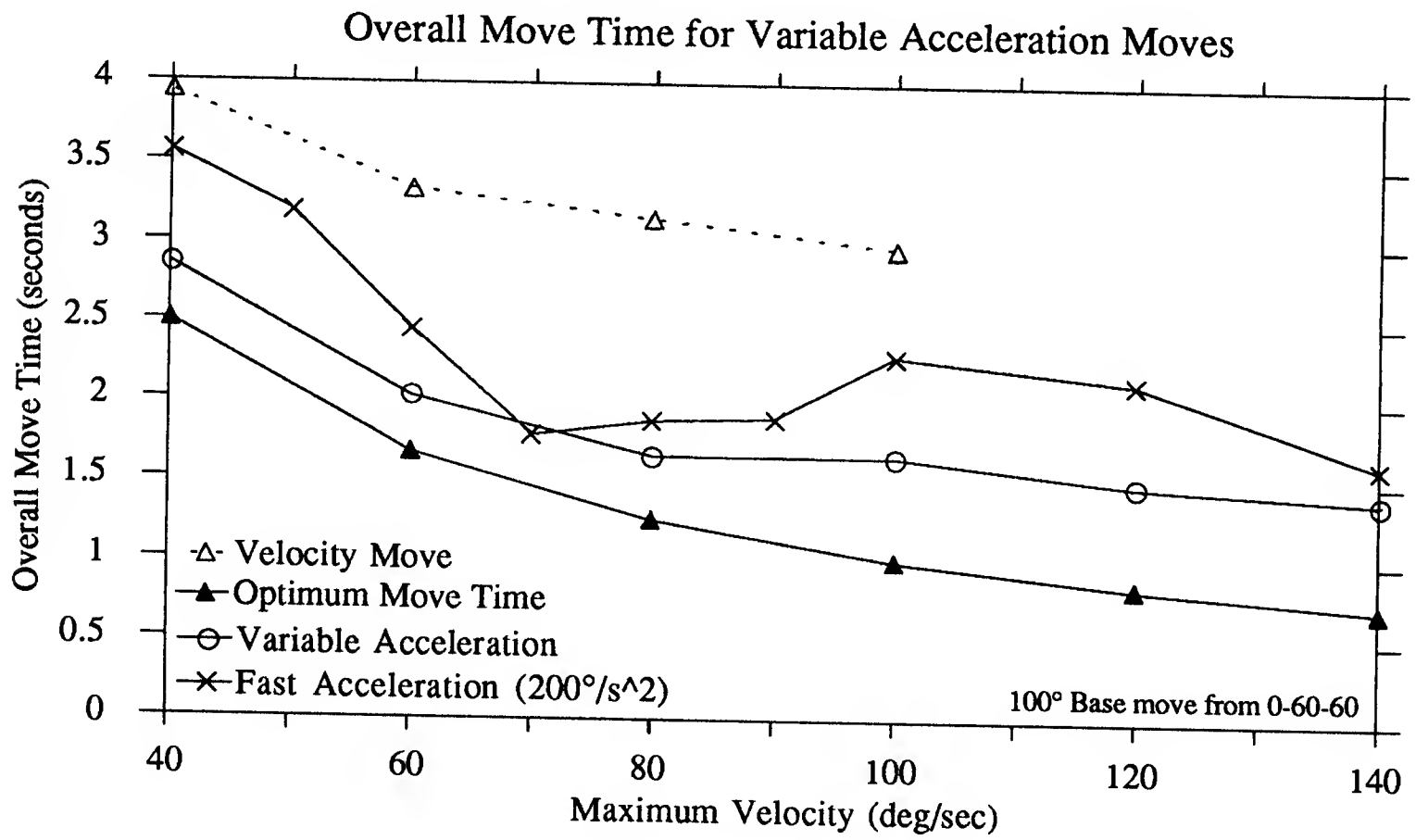


Figure 5.13: Variable acceleration move for a 100 degree move of the base joint with the first and elbow axes at 60 degrees apiece. The variable acceleration is compared against the constant 200 deg/sec^2 acceleration move.

completed; as usual, the "Optimum Move Time" refers to a rigid arm. The lower graph shows the settling time required for the vibrations to damp out. The amplitude of the residual vibration after the move has stopped correlates with this settling time.

At first glance the prefiltering techniques seem to offer little advantage over the traditional velocity or acceleration commands. Look carefully: the maximum velocity of the prefiltered techniques goes up to 300 degrees/sec. But the filters do not cancel vibration out exactly. More precisely, the filter is canceling out the *first* mode of vibration, but is still exciting the *second* mode of vibration, which is vibrating at 6.0 Hz and 0.1 damping.

The prefilter frequency was set to 2.75 Hz by making a plot of a velocity move and measuring the period of the oscillation. This calculation is sufficiently accurate for our purposes, but what if we were really off in our estimate of the frequency?

For comparison purposes we tuned a pair of prefilters to 3.4 Hz (25% higher than 2.75 Hz) and repeated the experiment. Figure 5.15 shows the same 100 degree move in the base joint with a two impulse prefilter and a three impulse prefilter deliberately "untuned". The dip in the settling time of the untuned three impulse filter at 220 deg/sec really does exist: it is an example of the phase of the residual 2.75 Hz vibration matching well with the move time.

The lower graph of Figure 5.15 has the settling time from both prefilter frequencies plotted on the same graph. It is evident that the two impulse filter is not robust to errors in the estimate of the natural frequency. However, the three impulse sequence gives virtually the same results with the tuned filter and the untuned filter. It is also worth noting that the tuned two impulse filter overall move time is approximately the same as the overall move time for the three impulse filter. The two impulse filter finishes the commanded move 0.182 seconds earlier but has a greater amount of residual vibration from the second mode. The extra time spent waiting for the vibrations to damp out leaves the overall move time of the two techniques to be

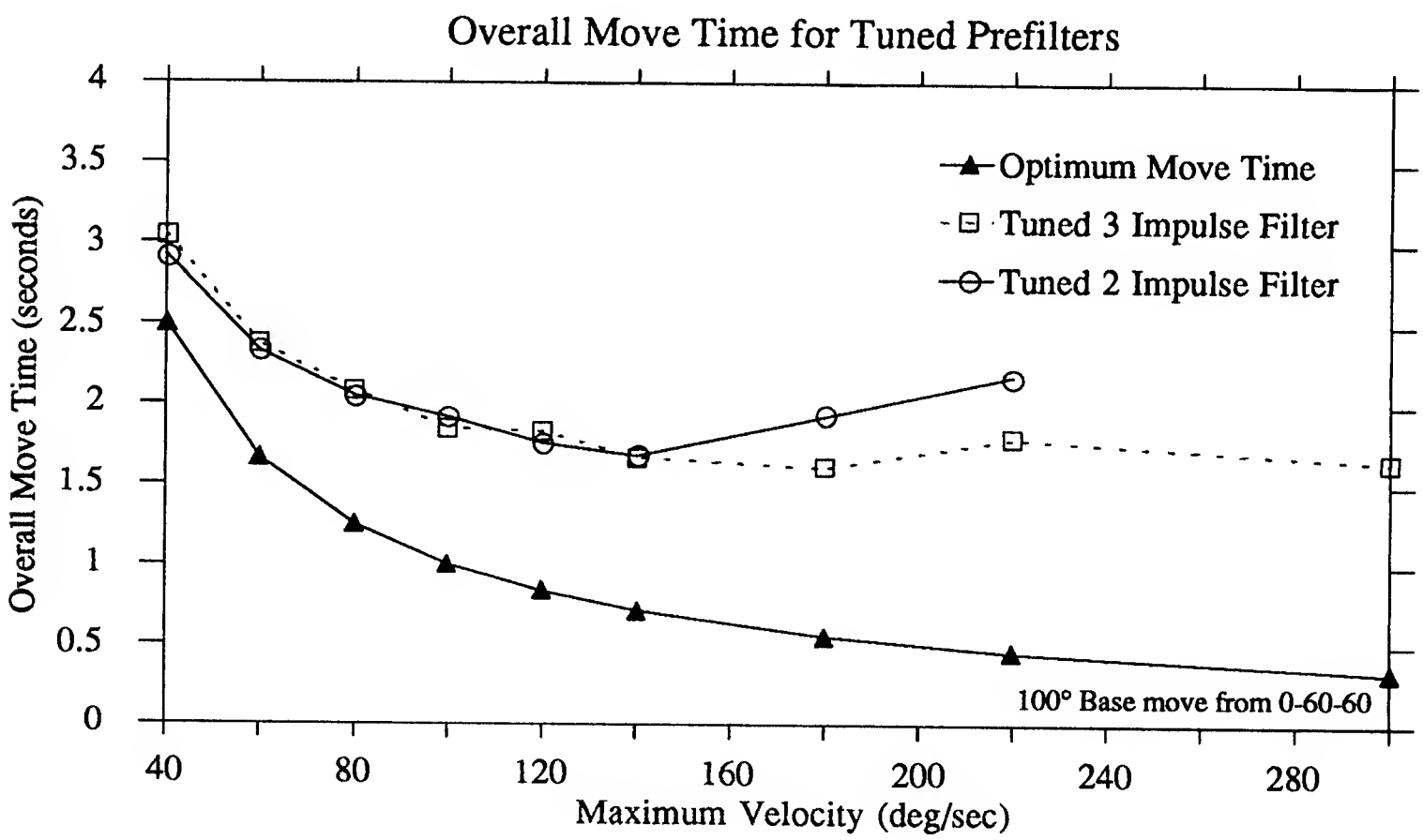
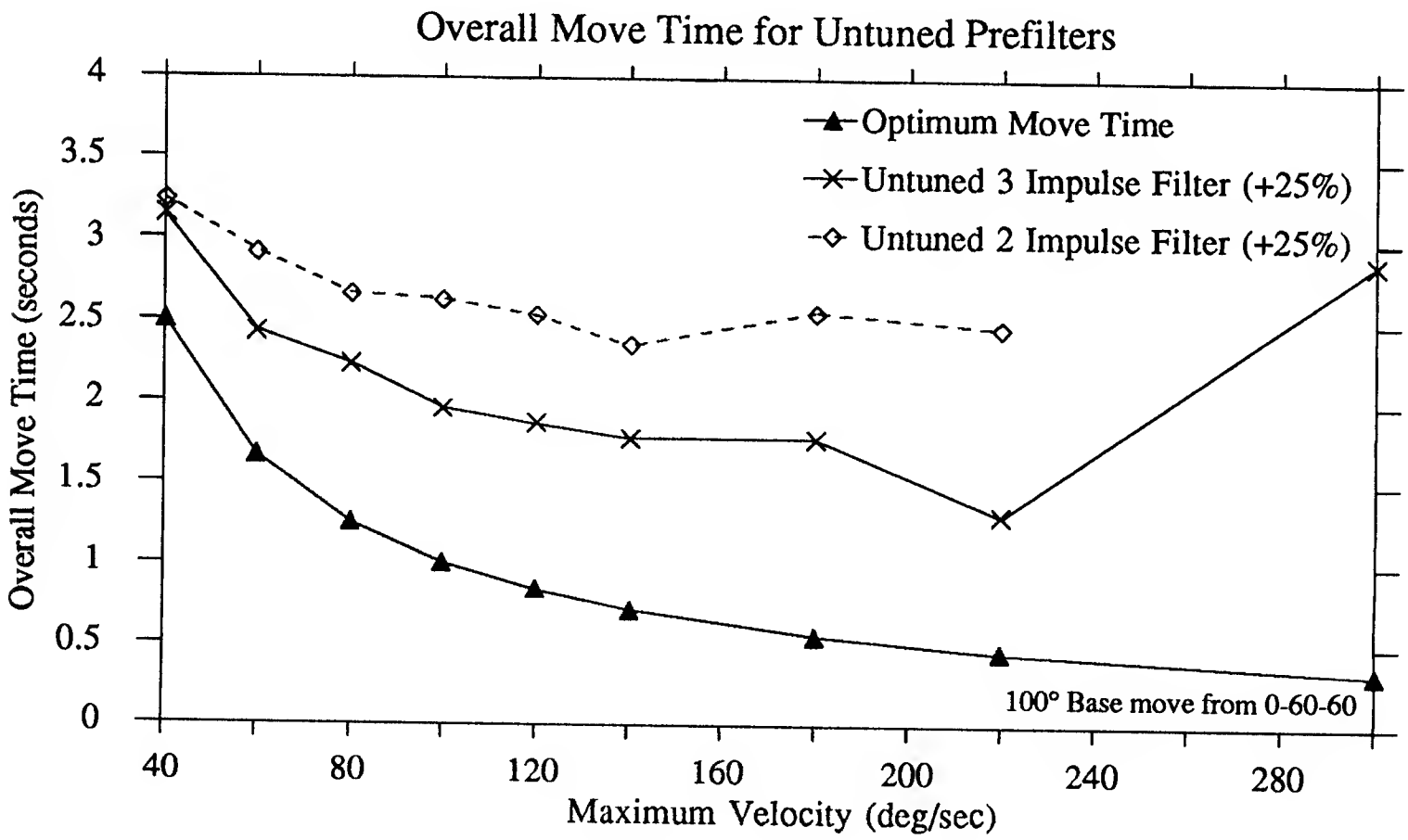


Figure 5.14: Simple prefiltered moves for a 100 degree move of the base joint with the first and elbow axes at 60 degrees each. The prefilters are tuned to 2.75 Hz and 0.10 damping. Settling criteria is a base joint peak to peak vibration amplitude < 0.10 degrees.



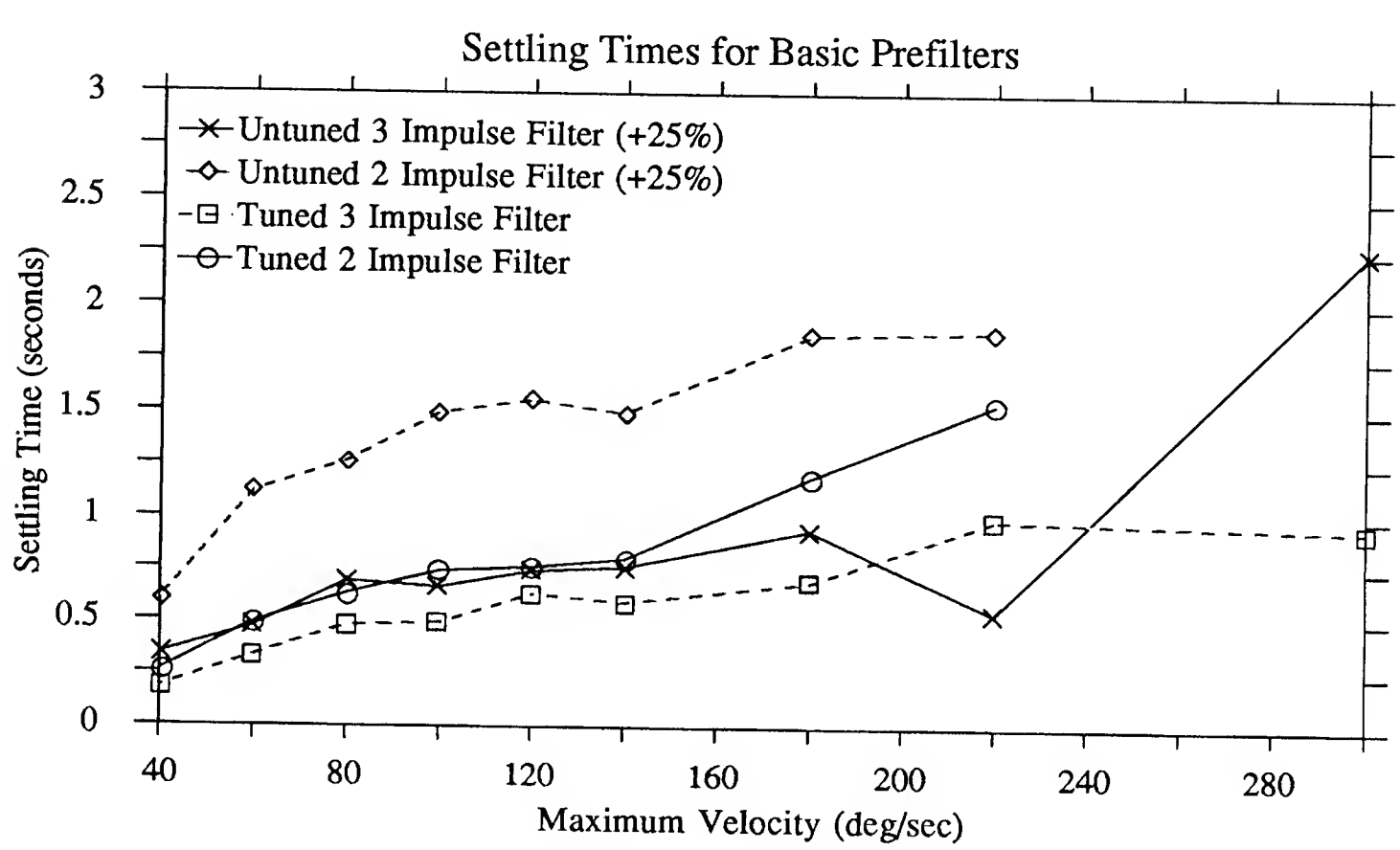


Figure 5.15: Simple prefiltered moves for a 100 degree move of the base joint with the first and elbow axes at 60 degrees each. The upper graph shows two and three impulse prefilters deliberately tuned 25% high to 3.4 Hz and 0.10 damping. The lower graph compares the settling times for both the tuned and the untuned prefilters.

approximately the same.

5.3.3 More complicated prefilters and a general comparison of methods

The primary problem with the tuned three impulse filter is vibration from the second natural frequency of 6.0 Hz. A natural step to try to handle this vibration is to use an impulse sequence that is tuned for 2.75 Hz convolved with an impulse sequence tuned for 6.0 Hz. As mentioned in Section 5.1.2, this results in an impulse sequence which cancels vibrations out at both frequencies. Figure 5.16 compares the vibrations from two such filters. The upper graph shows the overall move time required for the longer filter sequences and the lower graph shows the vibration settling time. A "2 x 3" impulse filter is a pair of 3 impulse filters that have been convolved together. A "2 x 2" impulse filter is a pair of 2 impulse filters that have been convolved together. Please note that the maximum velocity tested is higher than in previous figures; the longer filter sequences make it possible to run the arm extremely fast without much residual vibration.

Section 5.3.1 inspires another way to try to reduce residual vibration and improve move times. If you recall, we selected the time required for acceleration to be just the length of a full cycle of vibration. This is the equivalent of a simplistic 2 impulse prefitler that doesn't take into account damping and modeling uncertainties. However, it does have the nice property that the acceleration is constant while getting up to speed and decelerating from speed.

We can improve on the behavior of the acceleration move by using a tuned prefilter. In this case, we use a three impulse prefilter tuned to 2.75 hertz, 0.1 damping. We select accelerations exactly twice the value that Equation (5.2) suggests: this gives an unfiltered acceleration time of just half a cycle of vibration. If we filtered it with the simplistic 2 impulse prefilter, we would end up with exactly the same

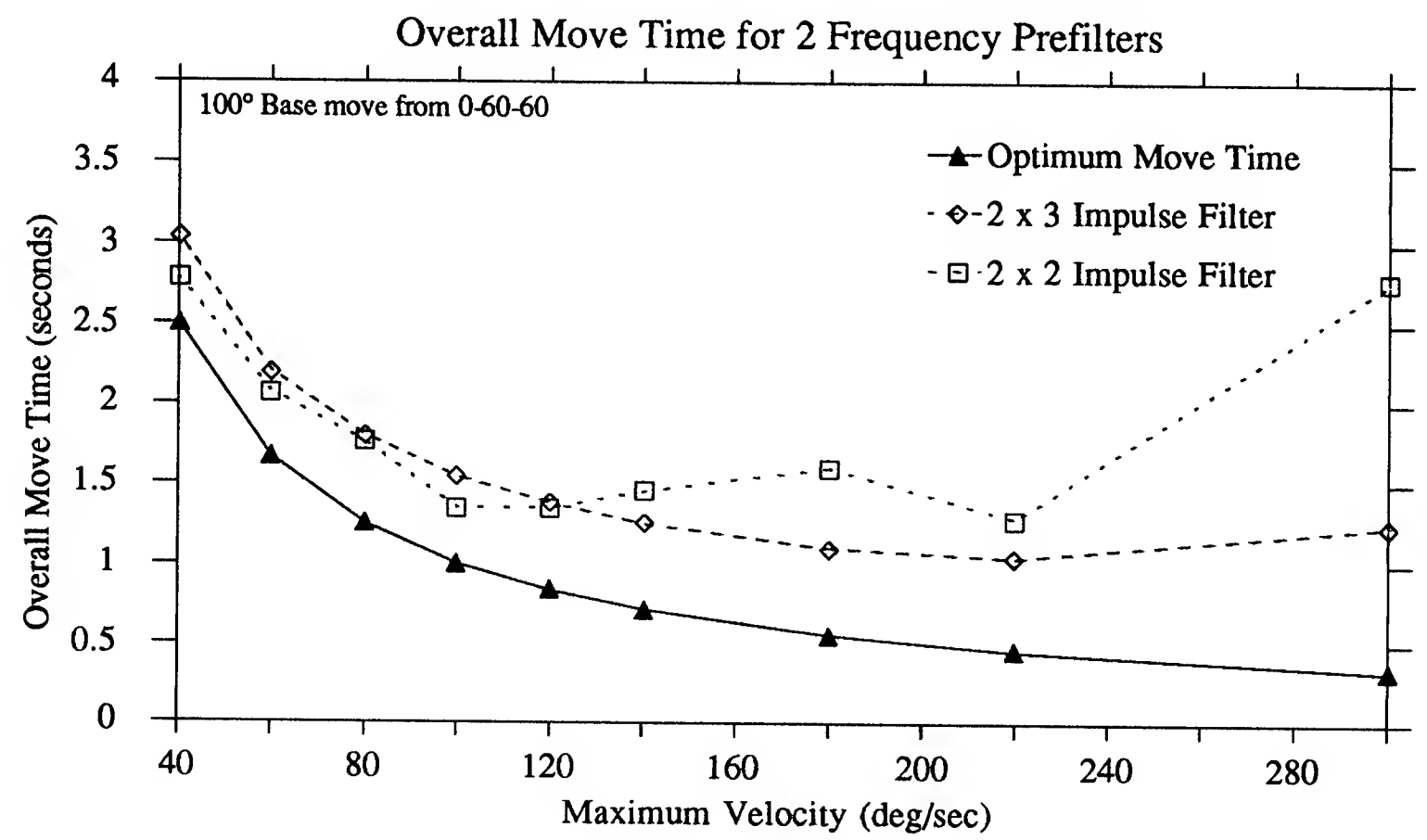


Figure 5.16: Two frequency prefilters for a 100 degree move of the base joint with the first and elbow axes at 60 degrees. The prefilters are tuned to 2.75 Hz and 6.0 Hz with 0.1 damping. The upper graph shows the overall move time required; the lower graph shows the settling time.

accelerations used in Section 5.3.1. However, for these tests we filter it with the longer three impulse prefilter that takes damping into account. The results of these tests are shown in Figure 5.17.

Figure 5.17 clearly shows that the filtered variable acceleration move has a much lower settling time than the simplistic acceleration move. It works extremely well up to 180 deg/sec maximum velocity. It does not cancel out all of the vibration at the high speeds because the accelerations are so high that they are exciting the 6.0 hertz frequency of the arm.

It is only fair to finish off this section with a comparison between all of the methods of vibration control. Figure 5.18 shows the move data from the standard velocity move, the tuned single frequency impulse filters, the two frequency filter, the variable acceleration move, and the filtered variable acceleration move. The upper graph compares the overall move time versus maximum velocity. The lower graph shows the settling time for each type of move. The two best techniques are the filtered variable acceleration and the two frequency filter, giving a minimal amount of residual vibration and a low overall move time.

One is tempted to ask if one can get the residual vibration even lower at higher speeds. The answer is certainly yes if you are willing to accelerate slowly and have very long, carefully tuned prefilters. But there is a point of diminishing returns: sooner or later the time gained by diminishing the residual vibration is offset by the time lost in slow acceleration and filtering out the vibration. Figure 5.19 is the same pair of graphs as Figure 5.18 with "Net Velocity" substituted for "Maximum Velocity."

I define "Net Velocity" as the distance the robot travels divided by the time it would have taken a rigid robot to make the commanded move. That is, the time the move would take if the settling time was zero. Let me use the two moves shown in Figure 5.9 on page 107 as an example. The 60 deg/sec velocity move command takes

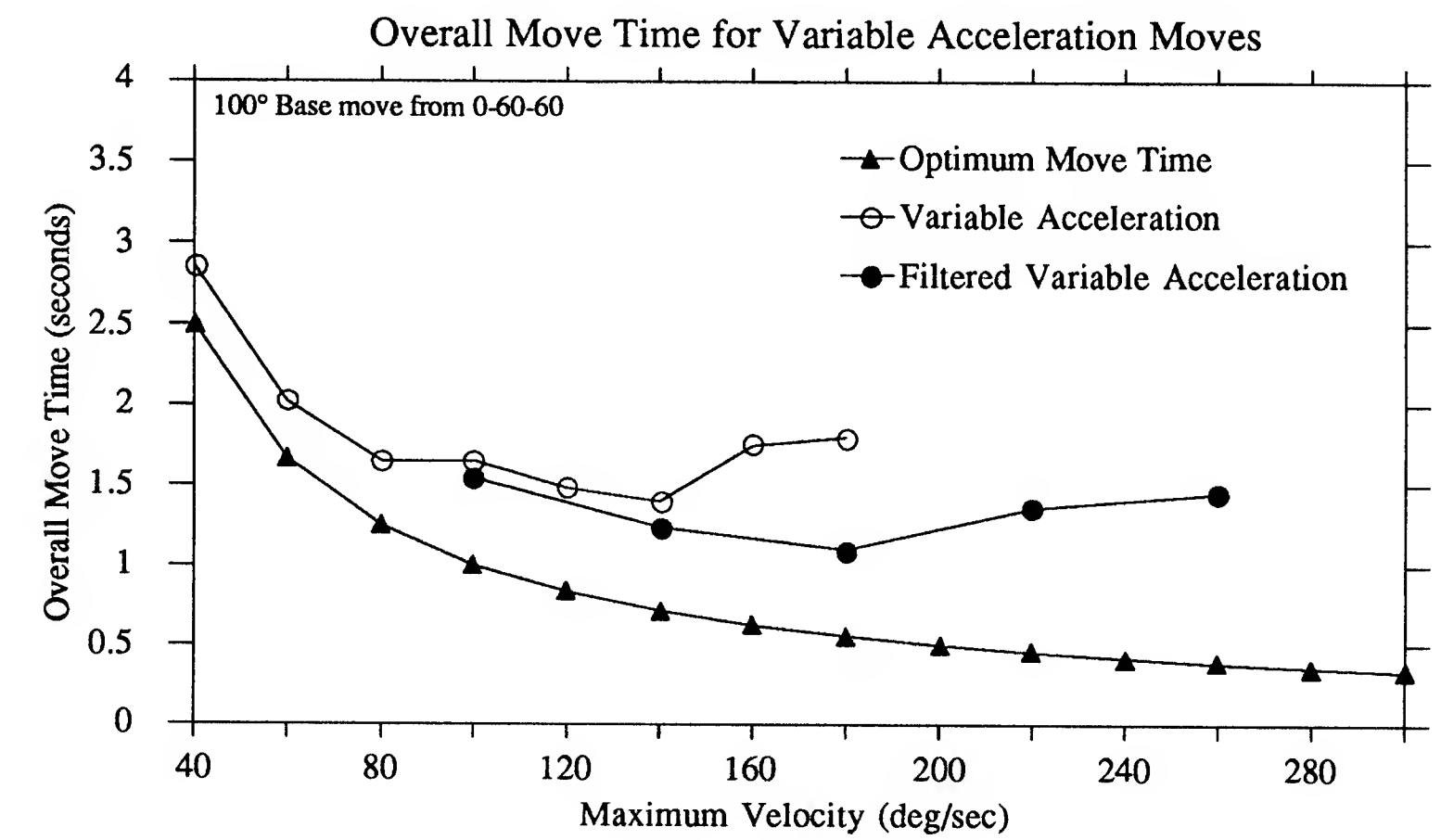


Figure 5.17: Filtered variable acceleration for a 100 degree move of the base joint with the first and elbow axes at 60 degrees. The prefilter is tuned to 2.75 Hz and 6.0 Hz with 0.1 damping. The upper graph shows the overall move time required; the lower graph shows the settling time.

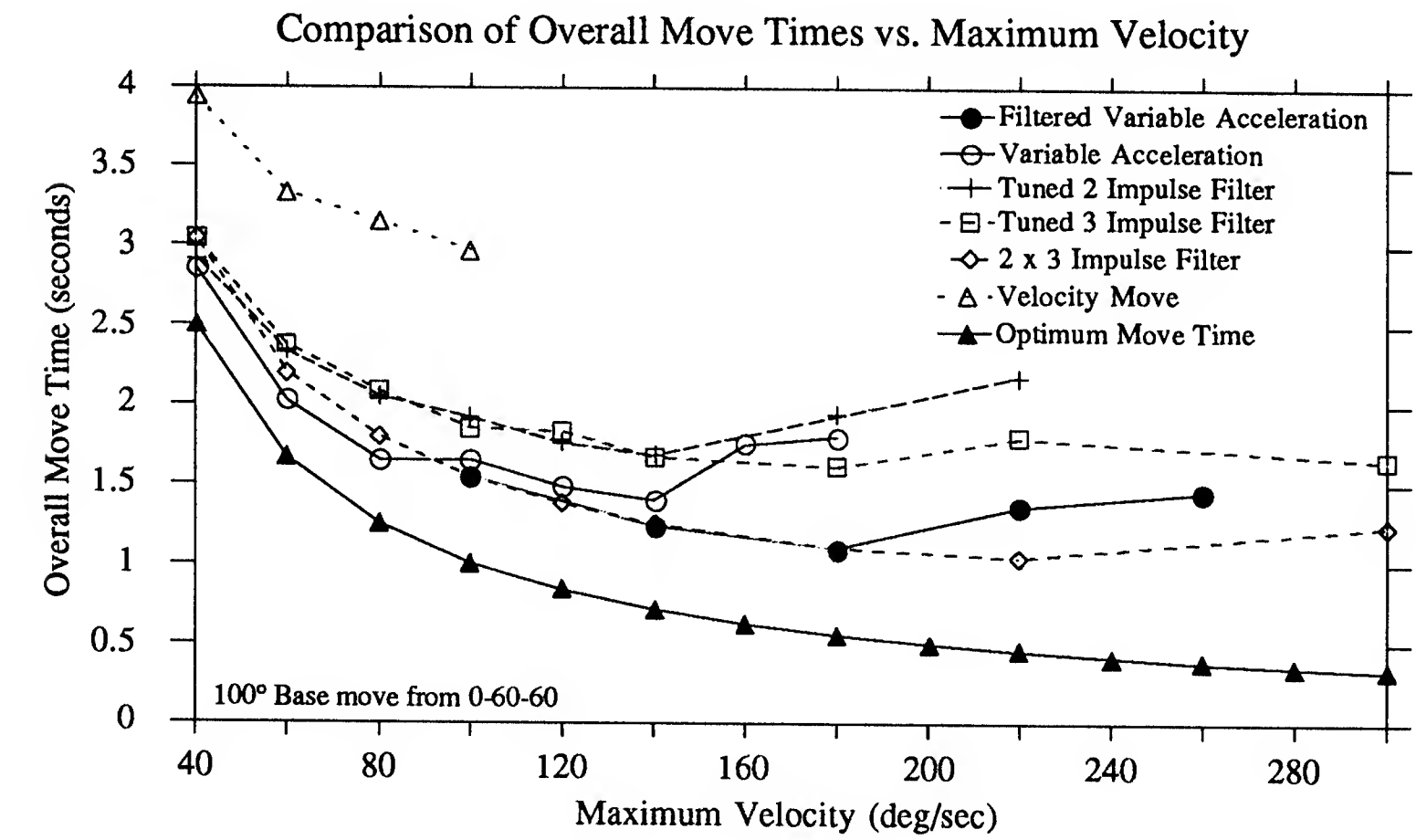
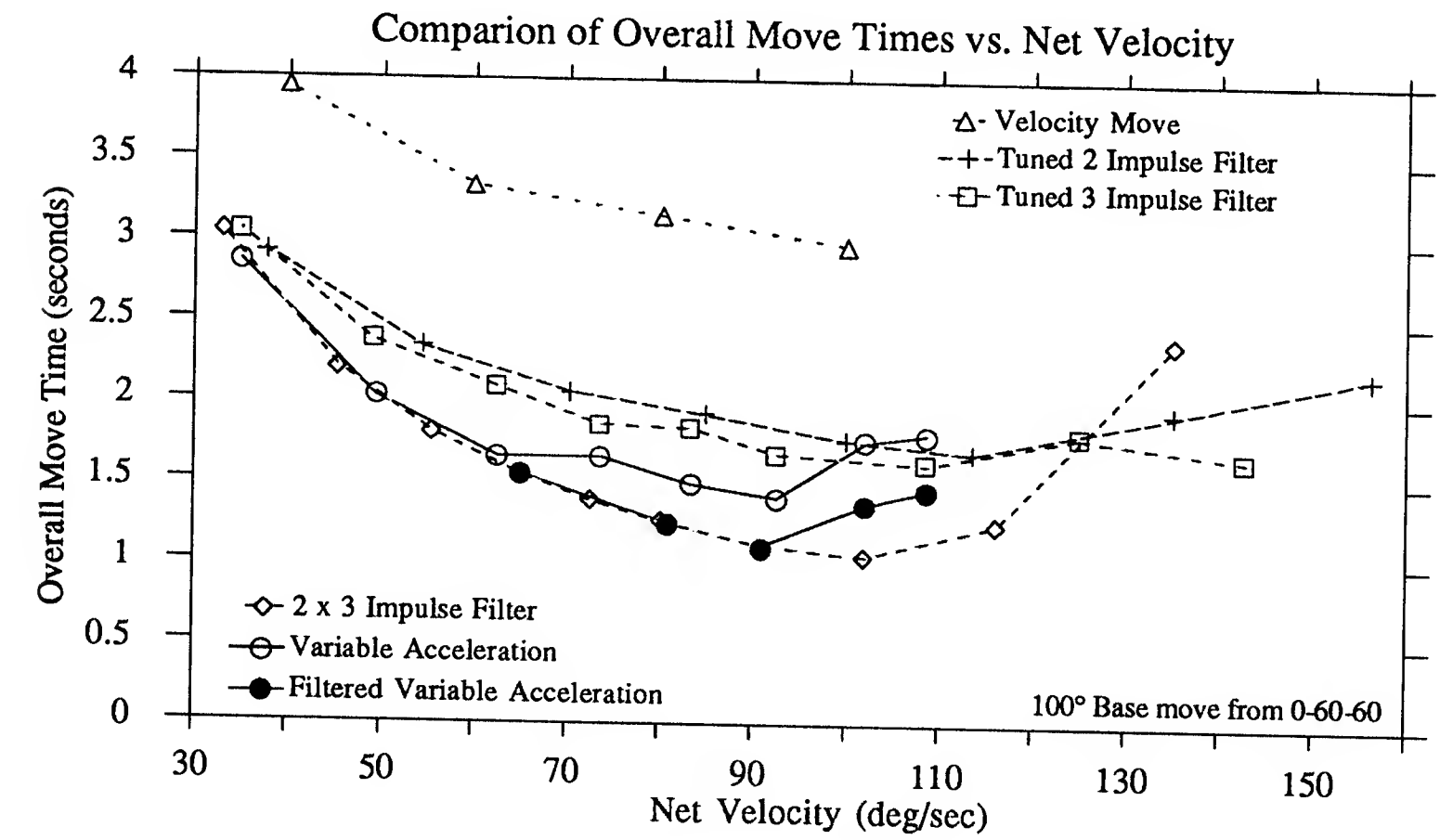


Figure 5.18: Comparison of the best options for the 100 degree move. The elbow and first joints are at 60 degrees.



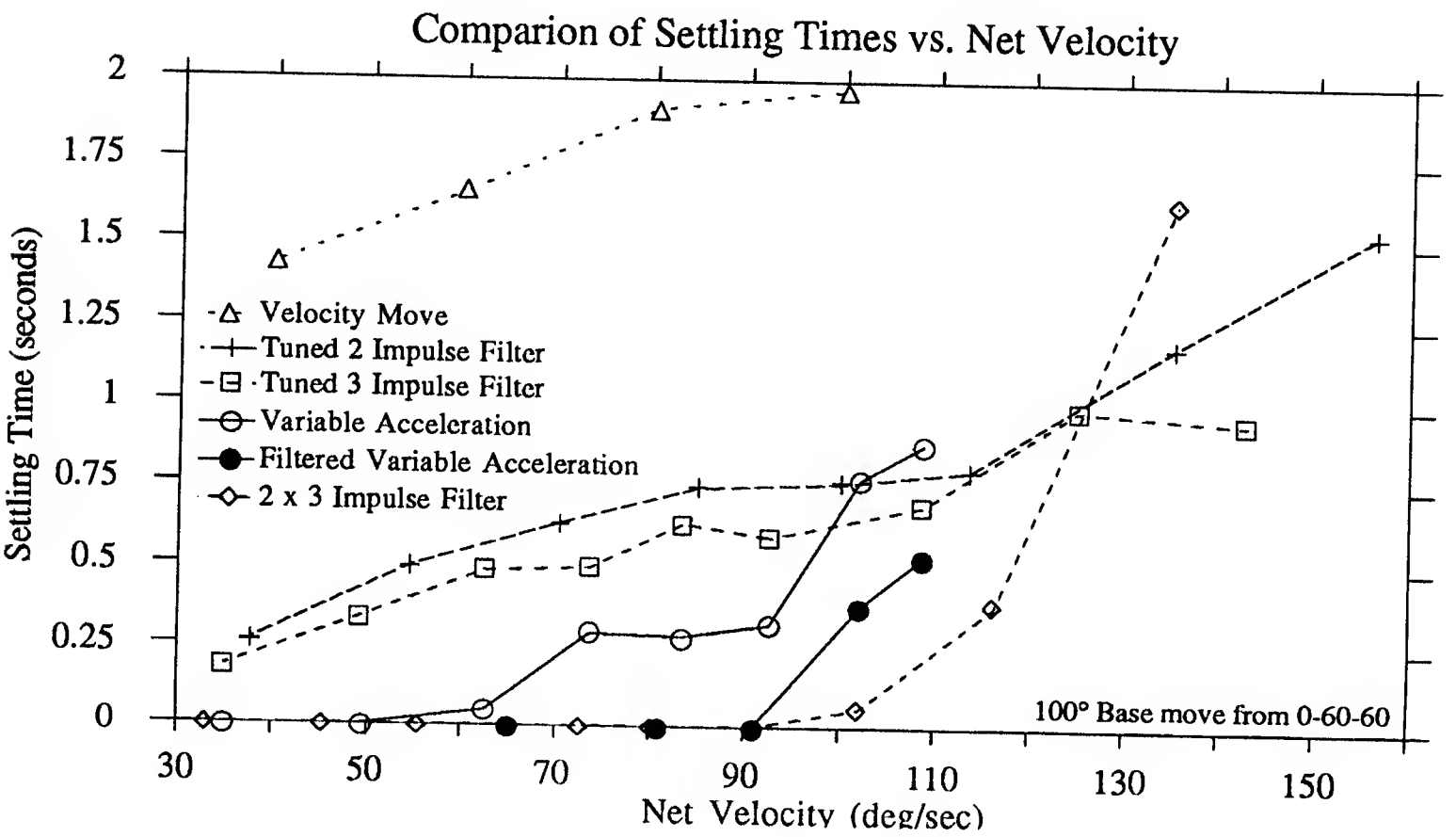


Figure 5.19: Comparison of the best options for the 100 degree move against the net velocity of each move. The elbow and first joints are at 60 degrees.

1.667 seconds to go from the start to the finish. Hence the net velocity is 100 degrees divided by 1.667 seconds or 60 deg/sec. The prefiltered command reaches a maximum speed of 180 deg/sec, but it has an inherent time delay of 0.364 seconds because of the prefilter. The commanded time in this case is $0.555 + 0.364$ seconds or 0.919 seconds. Hence the net velocity is 100 degrees divided by 0.919 seconds or 108.8 deg/sec. This is considerably slower than the maximum velocity of 180 deg/sec.

Net velocity is a useful concept because two moves with the same net velocity get to the finishing point at the same time regardless of prefilters or accelerations. Figure 5.19 shows that at the higher net velocities it is actually better to not use the complicated prefilters. In other words, if you have a strict time requirement on how long you can take making the move, then to minimize the residual vibration you would probably use the 3 impulse prefilter. On the other hand, if your goal is to minimize the overall move time, you would use either the filtered variable acceleration or a pair of tuned 3 impulse filters. You would probably not find it advantageous to use a higher maximum velocity coupled with a more complicated prefilter because the two effects tend to cancel each other out. A very complicated prefilter might be able to cancel out all vibration with a high maximum velocity, but its net velocity will be very slow and hence one of the simpler methods would be more effective.

5.4 Varying Vibrational Frequency Move

The previous section dealt with a relatively simple motion: swinging the base 100 degrees while keeping the elbow and first joints fixed. This has the nice property that the vibrational frequencies do not change as the arm moves, and that generally the only frequencies excited are the two out of plane frequencies. But the purpose of a complicated three degree of freedom robot is to try to control the vibrations in complicated moves. This section deals with the vibration control of a relatively nasty move that excites all four of the fundamental modes with an arm motion that

changes all of the fundamental frequencies.

The motion studied here is illustrated in Figure 5.20. The arm starts with the elbow and first joint at 15 degrees. Simultaneously each axis is moved 60 degrees, ending up with the base at 60 degrees and the elbow and first joint at 75 degrees. The moves are commanded in joint space so the payload of the robot does not travel in a straight line. The natural frequencies of vibration were experimentally measured for the starting and stopping positions. At the start position, the in-plane vibration is 2.2 Hz, $\zeta = 0.2$. The out-of-plane vibration is 3.0 Hz, $\zeta = 0.05$. At the finish position, the in-plane vibration is 2.6 Hz, $\zeta = 0.15$ and the out-of-plane vibration is 3.0 Hz, $\zeta = 0.15$.

Determining when a move was over was difficult because all three joints of the robot vibrate and they vibrate at different frequencies. We settled on a finishing criteria of 0.10 degrees peak to peak for the two joints at the base and a finishing criteria of 0.20 degrees peak to peak for the elbow joint. The criteria is not uniform because the elbow joint has a lower encoder resolution than the base joint: 0.018 degrees vs. 0.009 degrees. Actually, the disparate scheme works well since the elbow joint also has a lower servo stiffness. Most of the time the vibrations drop below the criteria on all of the joints at about the same time.

Figure 5.21 shows the move times for acceleration and velocity moves. The upper graph gives the overall move time required, the lower graph gives the settling time. We did not run test cases with the variable acceleration style moves from Section 5.3.1 because it was not clear that tuning the acceleration for any one of the four vibrational frequencies would improve the performance dramatically. Instead, we settled on showing two different acceleration rates and counting on their relatively slow accelerations (160 and 240 deg/sec^2) to reduce residual vibration.

Figure 5.22 shows what happens when you do simplistic three impulse prefiltering of the input command, which can be either a standard velocity command or an

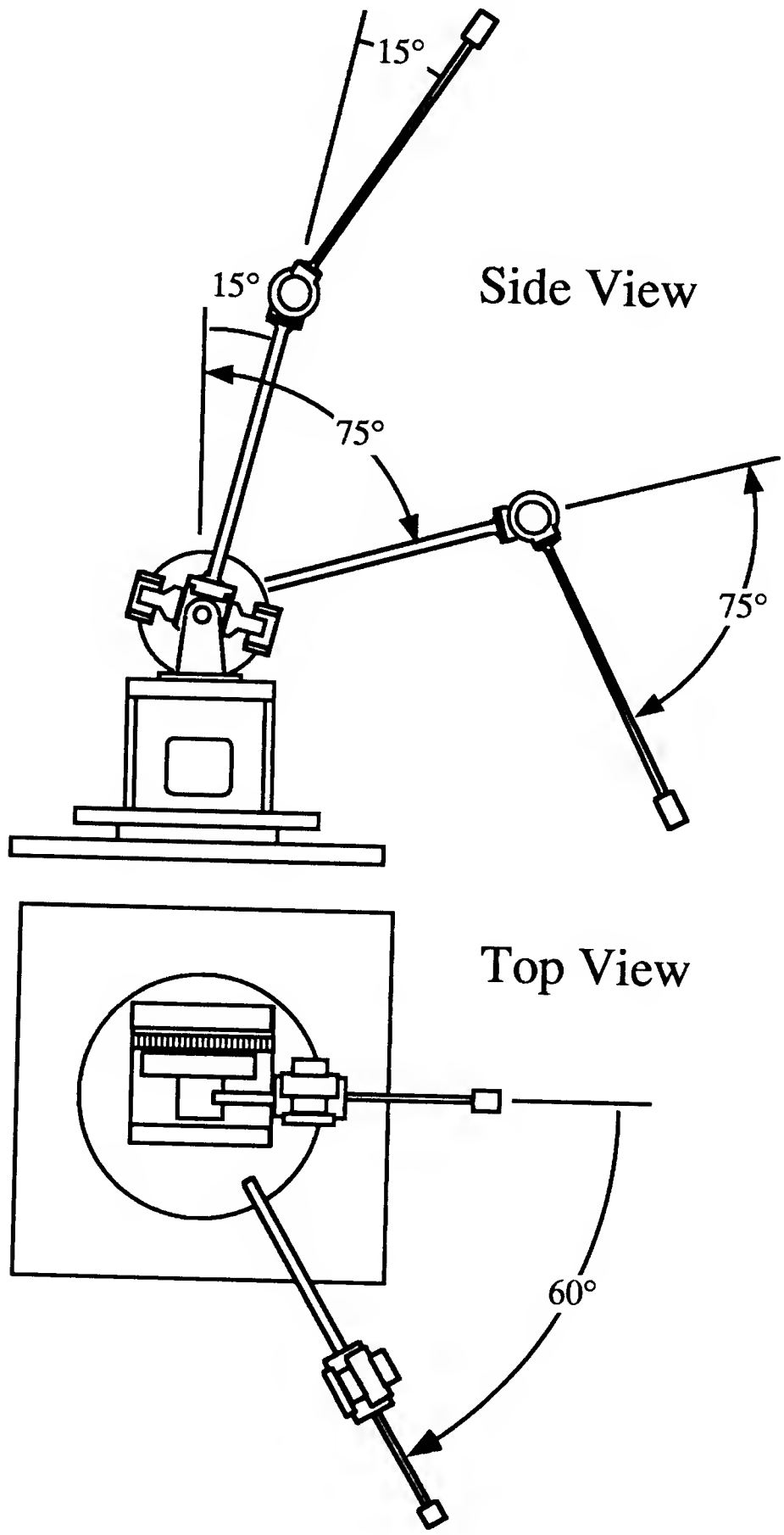


Figure 5.20: Configuration of the Flexbot for the varying vibrational frequency moves. The first joint and elbow joint move from 15 to 75 degrees. The base joint moves 60 degrees.

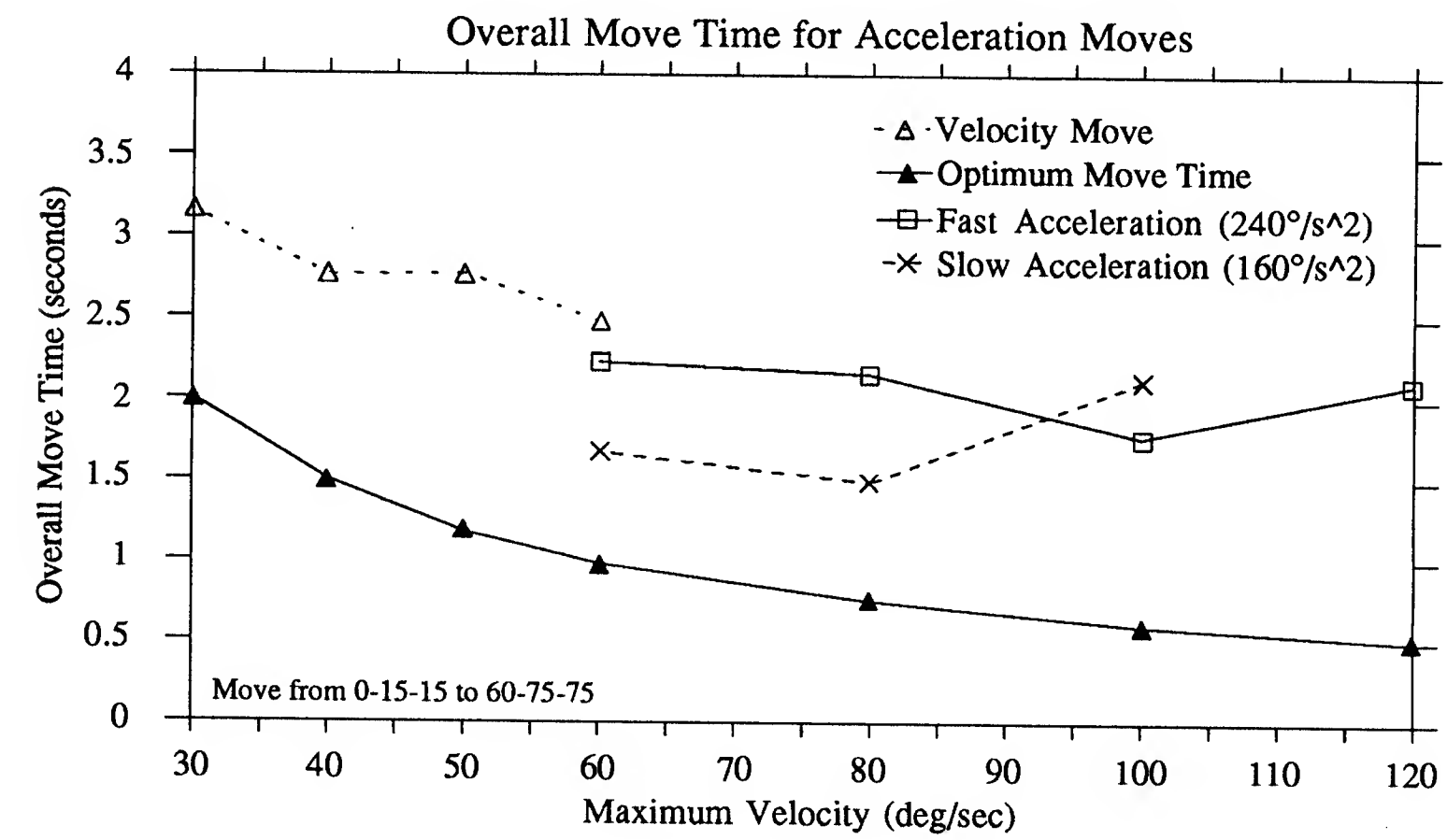


Figure 5.21: Different control methods used on the complex move from 0–15–15 to 60–75–75. The upper graph shows the motion with a velocity command and with slow accelerations. The lower graph shows the settling times.

acceleration command. The first prefilter was tuned to 3.0 Hz and $\zeta = 0.05$, which is the first fundamental frequency of vibration of the arm in the starting location for out-of-plane vibrations. The second prefilter used was tuned to 2.8 Hz and $\zeta = 0.15$ which is a compromise between the in-plane and out-of-plane vibrational frequencies at the end of the motion. Both efforts reduce the residual vibration, but they do not eliminate it.

In this case the residual vibration is not from higher frequencies. It is equal parts of in-plane and out-of-plane fundamental mode vibration. In an attempt to reduce this vibration, we tried prefiltering an acceleration move. We used a combination of the fast acceleration move (240 deg/sec^2) feeding into a 2.8 Hz, $\zeta = 0.15$ prefilter. The resultant motion is shown in Figure 5.22. The combination of an acceleration move and a prefilter gives a low amount of residual vibration.

The next thought that comes to mind is that perhaps we can improve the response of the arm the same way done in Section 5.3.3. I.e., convolve several filters together to cancel more frequencies. The vibration remaining after the simple prefiltered moves is at the lower frequencies of the arm. To try to minimize this vibration we use two prefilters convolved together. The first is tuned to 3.0 hertz, 0.15 damping and the second is tuned to 2.6 hertz, 0.15 damping. Figure 5.23 shows the results of the longer filter sequence. The "2 x 2" impulse filter uses a pair of two impulse prefilters tuned to the above frequencies. The "2 x 3" impulse filter uses a pair of three impulse prefilters tuned to the same frequencies.

Using two prefilters eliminates just about all of the vibration at the lower frequencies. However, as you run the arm faster you get vibration at higher frequencies, primarily at a frequency of 5.6 hertz with 0.075 damping. Our final attempt to take care of all of these frequencies is shown in Figure 5.23. We use *three* prefilters convolved together. Each filter has three impulses. They are tuned to 3.0 hertz/0.15 damping, 2.6 hertz/0.15 damping and 5.6 hertz/0.075 damping. As you can see, the settling

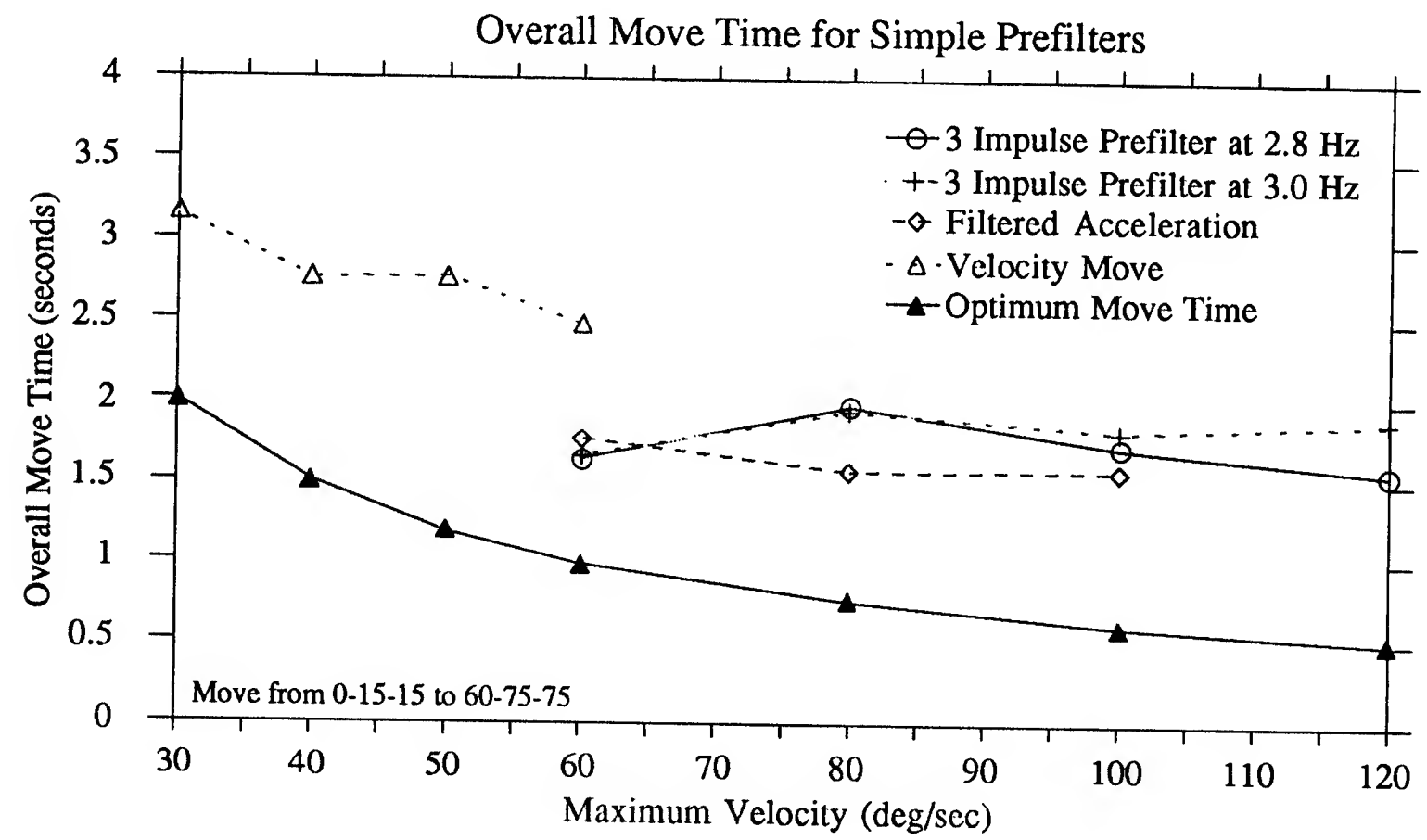
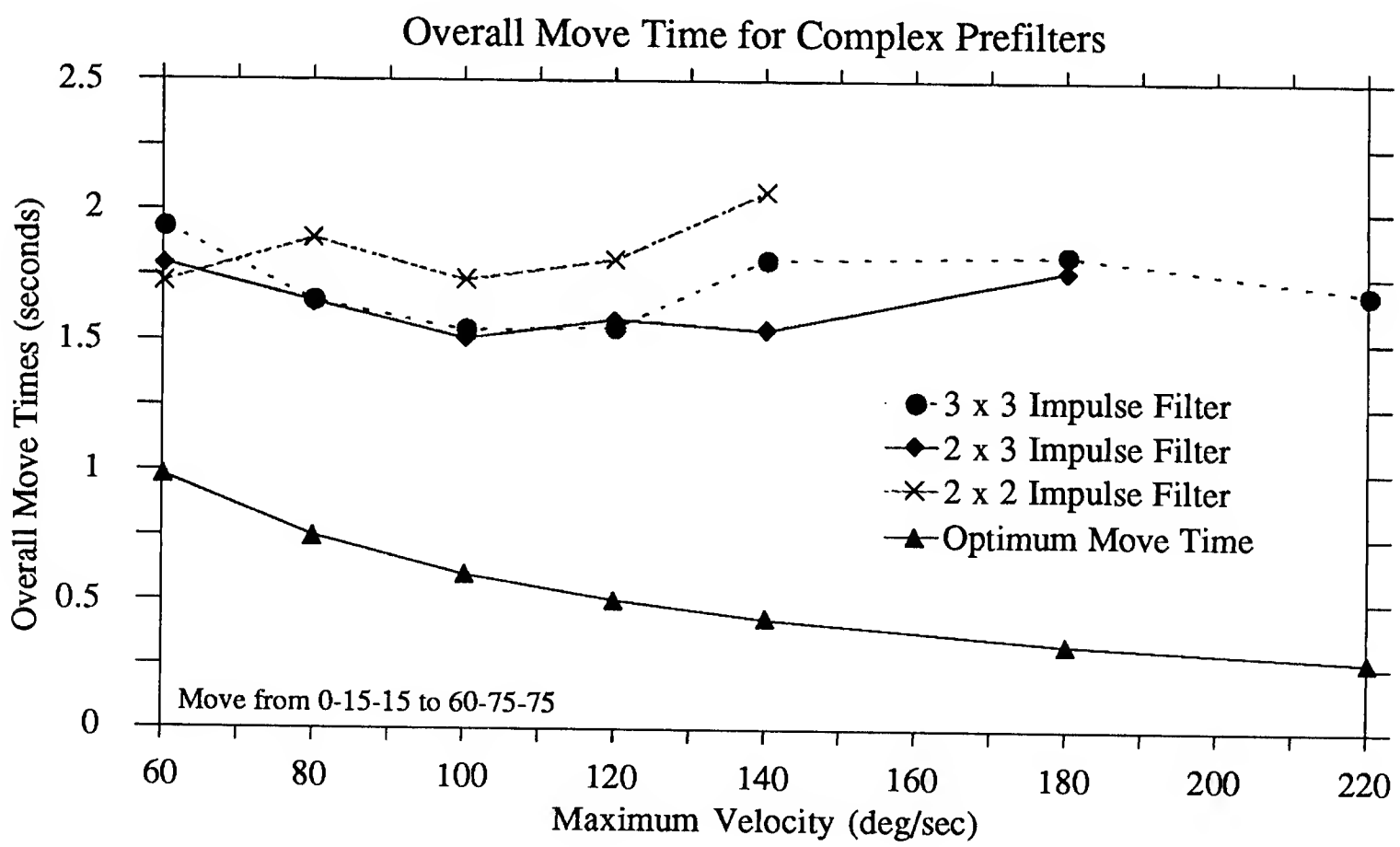


Figure 5.22: Simple filtering methods used on the complex move from 0–15–15 to 60–75–75. The filtered acceleration move is a combination of a three impulse prefilter at 2.8 hertz and an acceleration of 240 deg/sec^2.



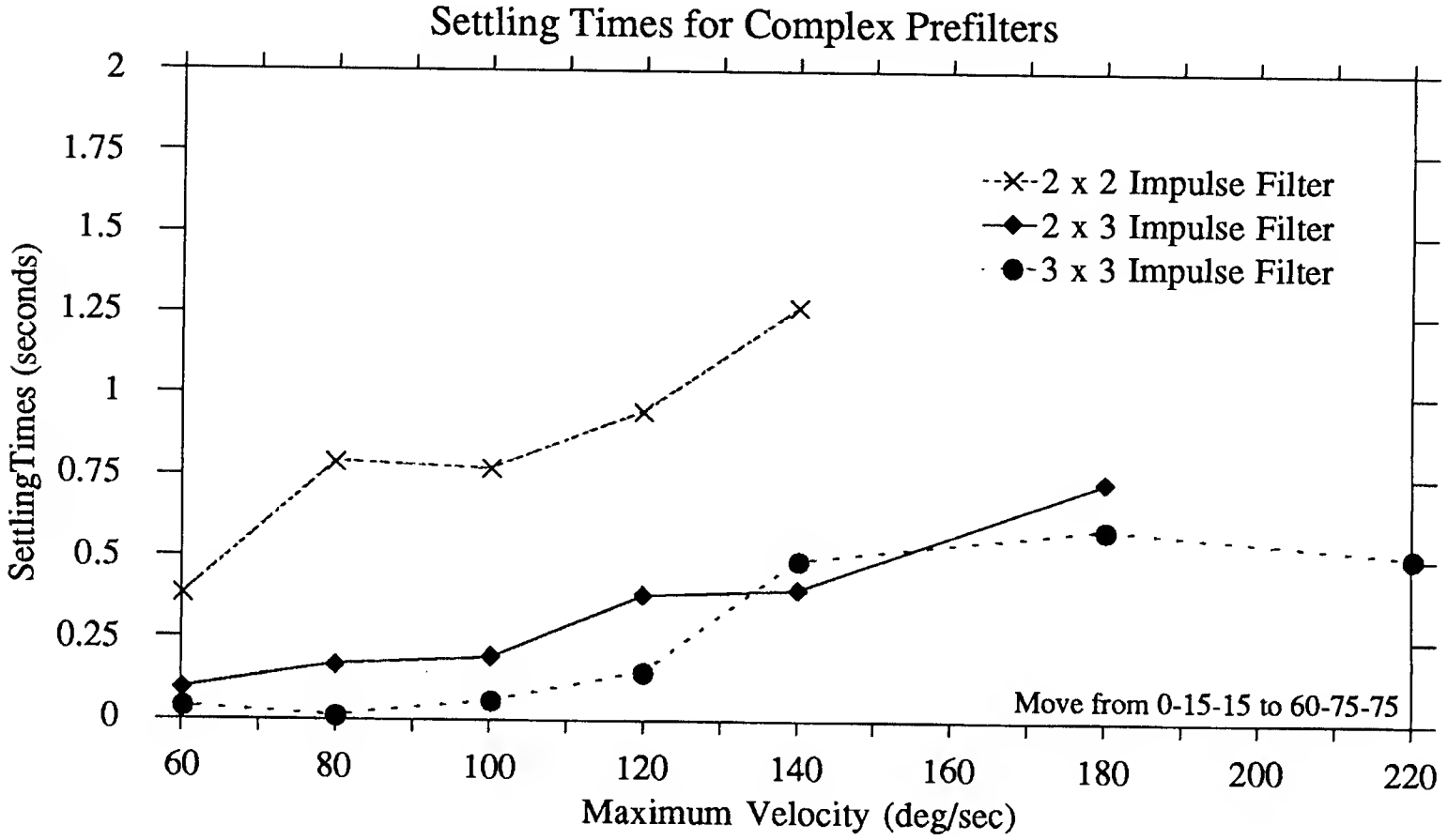


Figure 5.23: Complex filtering methods used on the complex move from 0–15–15 to 60–75–75.

time for this move is very low at the slower speeds. However, at the higher speeds we still get vibration.

Figure 5.24 gives a general comparison between the different vibration control methods. The upper graph compares the overall move time vs. the maximum velocity. The lower graph compares the settling time against the maximum velocity. It is evident that the fancy filtering techniques give a lower amount of residual vibration, but the overall time for moves remains relatively constant.

Figure 5.25 is the same comparison as Figure 5.24 but using the concept of net velocity explained in Section 5.3.3. The overall move time is suprisingly flat when plotted against the net velocity. With just about anything you do, you still end up with the best overall move time being approximately 1.5 seconds. The choice of vibration reduction technique is dependent on your criteria: if it is important to have no residual vibration, then you should move slowly with a complicated prefilter. If it is important to get to the desired position quickly and it doesn't matter that you have to wait for the vibrations to damp out, then you should use a simple prefilter to eliminate some of the vibrations and run the robot at a high speed.

5.5 Cartesian Motion

Robots are often operated in cartesian space instead of joint space. As described in Section 4.4.2, we implemented a simple cartesian motion capability on the Flexbot. The position of the payload is specified in cartesian space. As the robot moves it takes each desired cartesian position of the payload and uses the inverse kinematics of the robot to generate the correct joint positions. There is no attempt to deal with the dynamics of the system, so the position of the mass during a move tends to oscillate.

For our experiments we started with the robot at the cartesian coordinates of $(-25.0, 25.0, 10.0)$ and moved it 50 inches in the X-direction to $(25.0, 25.0, 10.0)$.

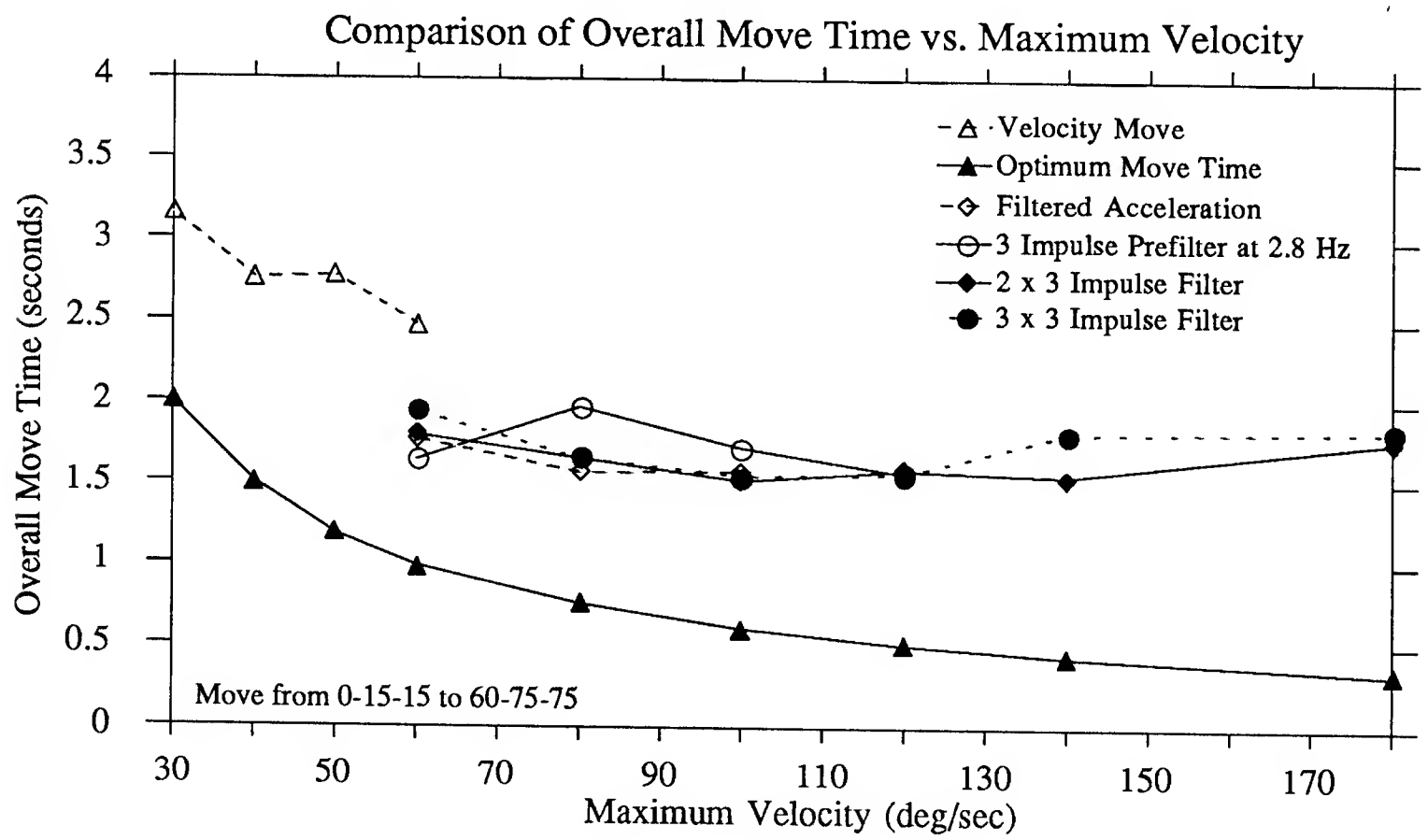


Figure 5.24: Comparison of the best methods for the complex motion from 0–15–15 to 60–75–75.

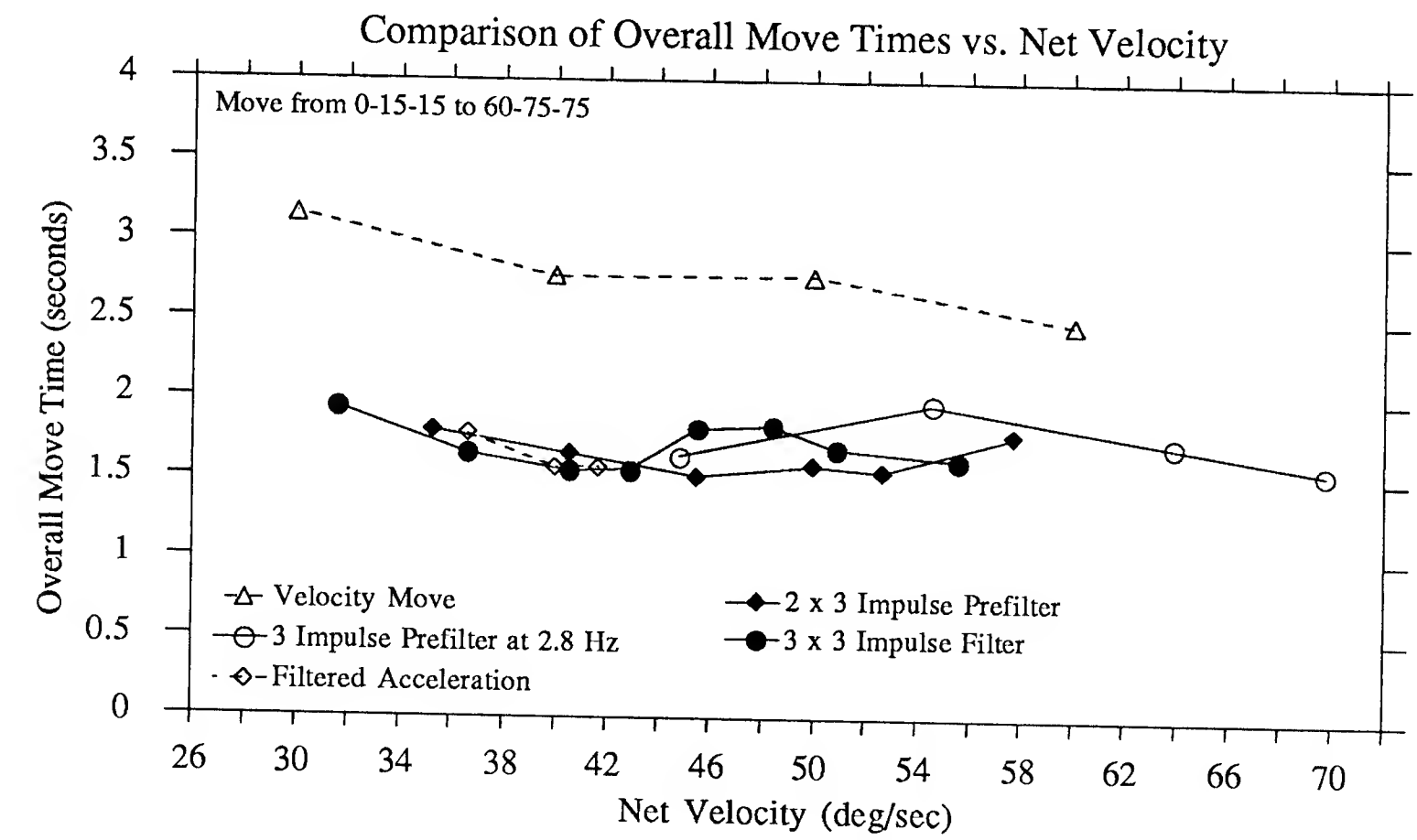


Figure 5.25: Comparison of the best methods for the complex motion from 0–15–15 to 60–75–75 against net velocity of the move.

This corresponds to joint coordinates of $\pm 45.0°, 33.88°, 82.19°$, where the order is base, first, elbow. The motion is symmetric about (0.0, 25.0, 10.0) where the joint positions are $0.0°, 13.05°, 112.97°$. A picture of the motion can be seen in Figure 5.26. Information of where the payload actually is comes from measuring the joint angles and running them through the kinematic equations. Hence our position estimate is rather inaccurate as it does not take into account the bending of the links. Unfortunately we have no better options at this moment, so the data will have to be accepted as preliminary results with a warning about the inaccurate method used to derive the position.

To give an idea of how well the robot does at tracking the desired path, Figure 5.27 has two graphs showing the path taken by the robot. Positions of the Y and Z coordinates are shown as X varies from -25 to $+25$; ideally Y and Z should remain constant throughout the move but you can see that they wander. In this case, the maximum velocity of the move was 60 inches/sec at the payload. The solid black line is the path recorded by a standard velocity move. The dashed line is the path recorded by an acceleration move with an acceleration of 200 inches/sec^2. Although the error in tracking the desired trajectory is not significantly different between the two moves, it is clear that the acceleration path is smoother.

Filtering is an interesting issue in cartesian motion. There is an ongoing debate on whether it is better to prefilter the motion in the cartesian space and convert it to joint angles or to convert the trajectory into joint space and perform the prefiltering there. If the prefiltering is done in cartesian space, the commands to the robot will stay on the path of the desired trajectory. If the prefiltering is done in joint space, the commands are not guaranteed to remain on the desired path. However, the reasoning goes that prefiltering in joint space might be better because the prefilter will legitimately remove the vibration, whereas taking a prefiltered cartesian path and moving it to joint space does not insure that the path will appear filtered in joint

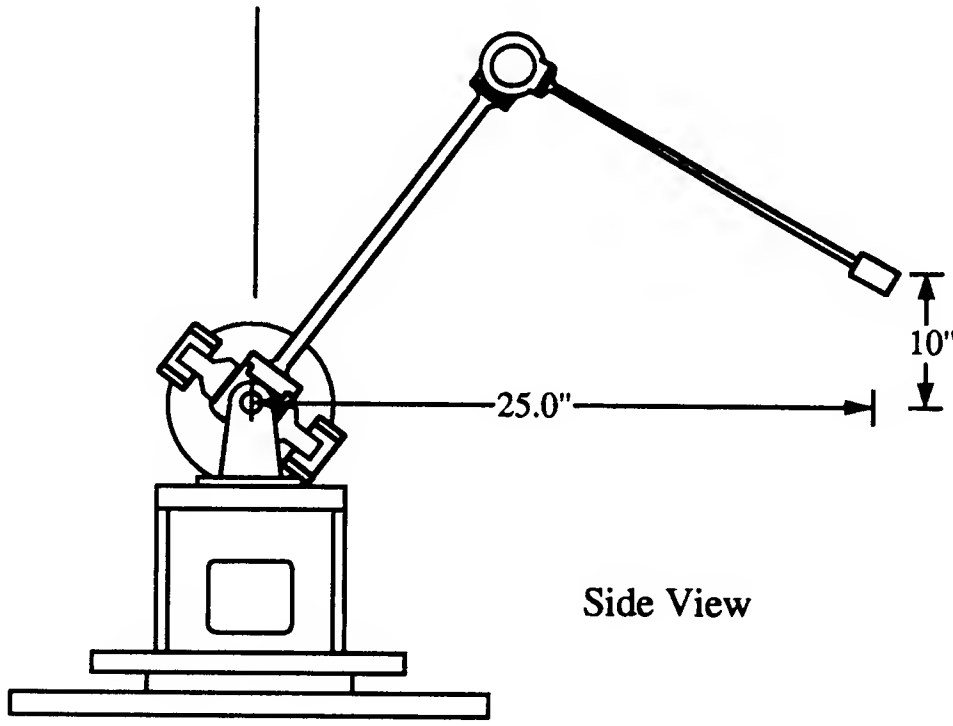
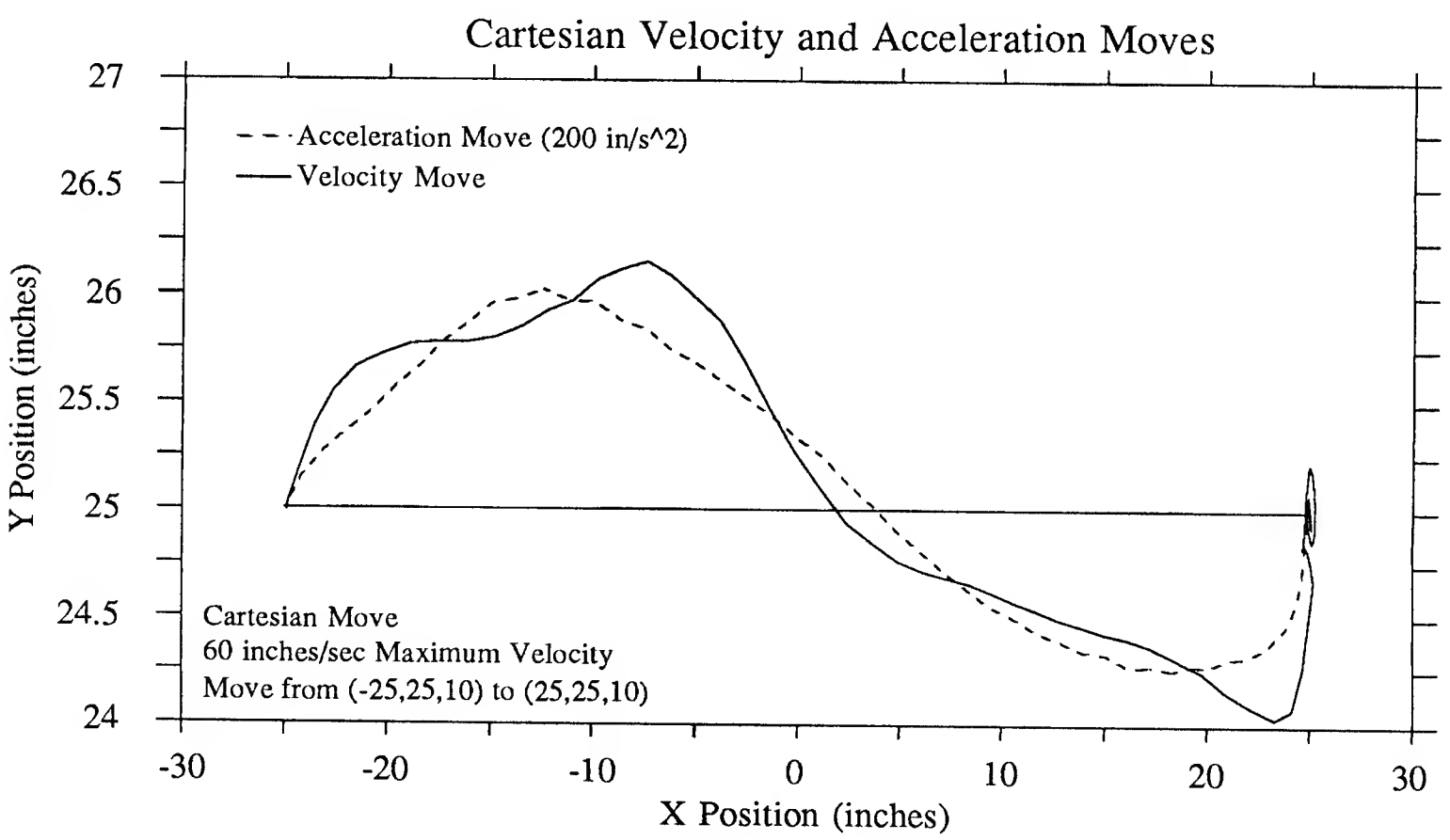


Figure 5.26: Configuration of the Flexbot for the cartesian motion. The payload of the robot moves in along the straight line between the start and stop positions.



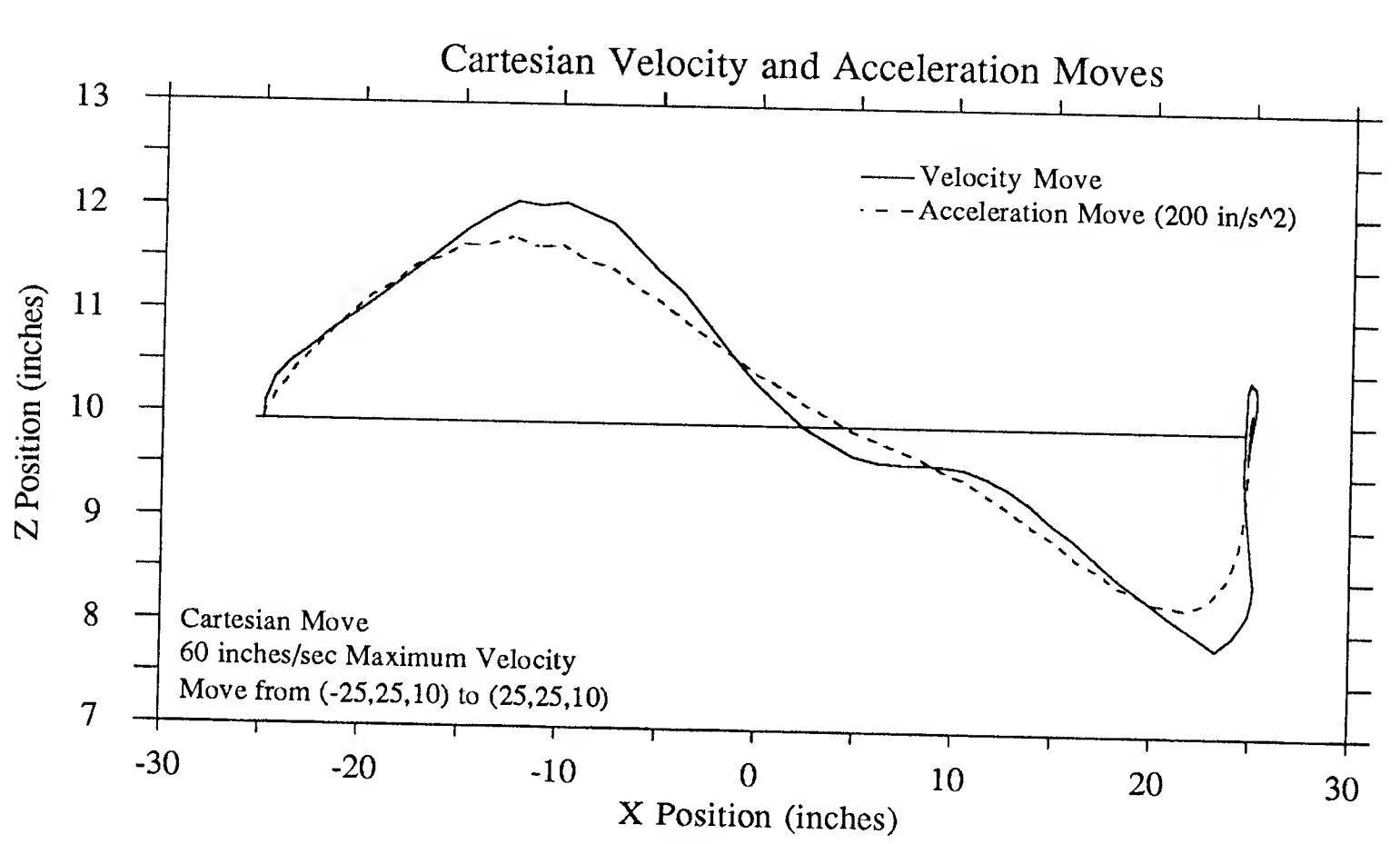


Figure 5.27: Trajectory followed by a velocity move and an acceleration move in cartesian space.

space. On the other hand, the linear motion from the cartesian space prefilter keeps the payload moving on a straight line. The elbow joint moves, but its amplitude of motion is smaller than when the filtering is done in joint space. Hence we might conclude that the smaller amount of motion might compensate for the nonlinear effects of filtering in cartesian space.

Figure 5.28 shows a comparison between a simple prefilter that was executed in cartesian space and the same prefilter executed in joint space. The prefilter is a three impulse filter tuned to 3.3 hertz, 0.1 damping. The cartesian space prefiltering does not affect the path the robot will follow: it is still told to move along the straight line connecting the starting point and the destination. The joint space prefilter warps the path that robot is being told to follow. This commanded path is shown as a dashed line in Figure 5.28. As you can see, prefiltering in joint coordinates induces an error of 0.75 inches off of the desired path for the Y position and just 0.2 inches for the Z position. The interesting comparison is that the distance between the commanded position for each of the move types and the actual position is roughly the same. That is, both types of filtering do about equally good jobs in keeping the end of the robot where it is being commanded to be. Of course, the cartesian filter is being commanded to be closer to where the operator really wanted the robot.

We turn to the issue of residual vibration. The nominal measure of residual vibration for cartesian space was the settling time of the vibration in the X direction, primarily because vibration was most pronounced in this direction. We used a 0.10 inch amplitude criteria to decide whether or not a joint had finished moving. Figures 5.29 and 5.30 are comparisons of overall move times and settling times for the different types of moves run at different speeds, using either a maximum velocity criteria or a net velocity criteria. As you can see, the cartesian space prefilter was marginally better than the joint space prefilter at keeping the vibration down.

There is a great deal more that can be done with cartesian motion: trying more

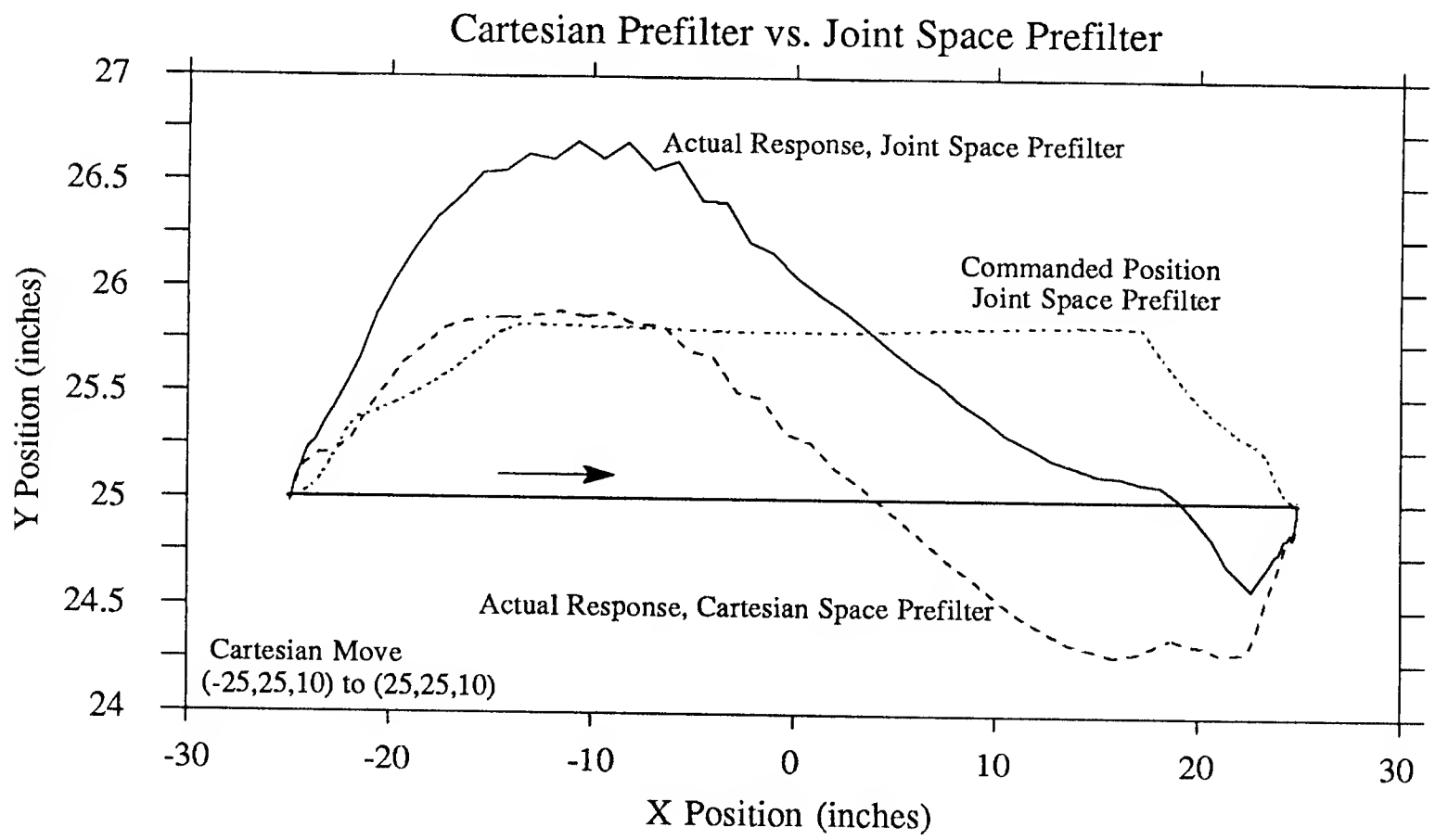


Figure 5.28: Prefiltered trajectories in cartesian motion: a comparison between joint space and cartesian space filtering.

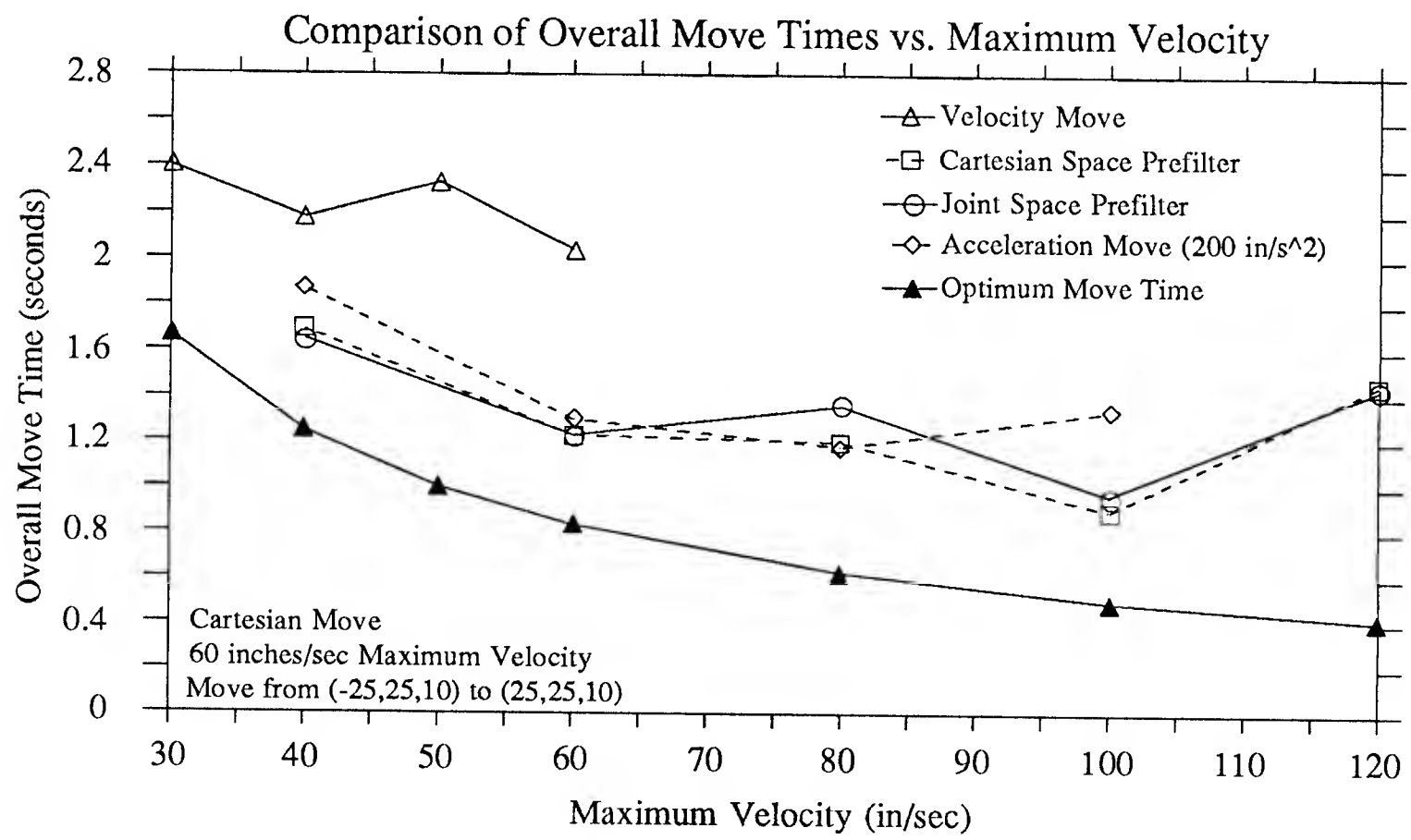


Figure 5.29: Comparison of different cartesian motion techniques.

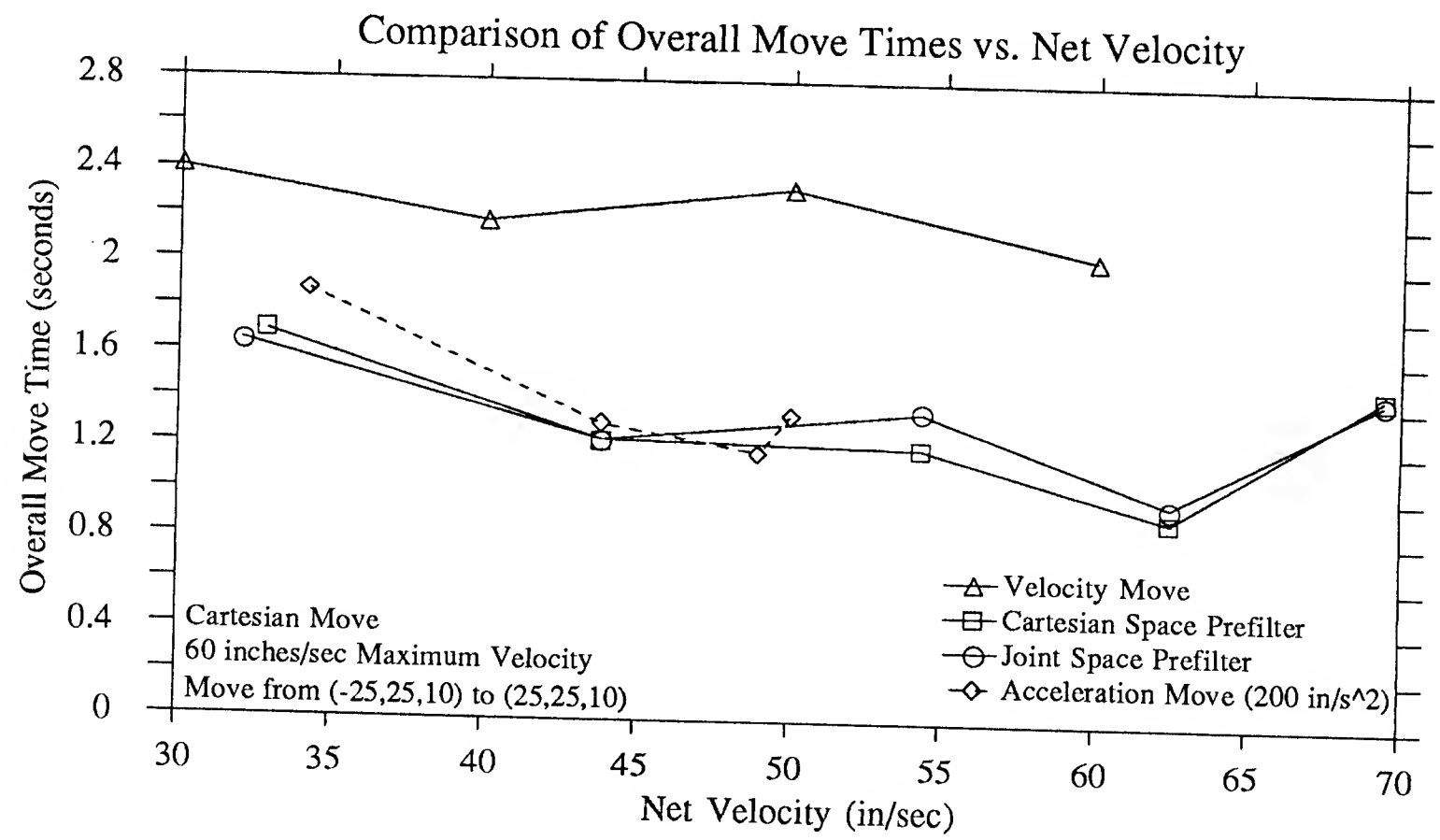


Figure 5.30: Comparison of different cartesian motion techniques against net velocity.

complicated filters, mixing cartesian and joint space filtering, or experimenting with complex trajectories and teleoperation. All this will have to wait until we develop a better method of determining just where the end of the robot actually is. At the moment we only have a very rough guess of how it is vibrating and where the payload is located; further experimentation calls for accurate recording of position data so we can determine what is really going on.

5.6 Conclusions

We have presented a mass of experimental data which demonstrates how the robot behaves with different moves and different positions of the arm. We have presented a number of simple techniques that control residual vibration with varying success. There are a few general conclusions and suggestions that can be drawn from this data.

First, there is a very important distinction between "long" and "short" moves. We did not deal with short moves. Short moves for this robot are best handled by gently moving the robot at a slow speed and a low acceleration. They avoid inducing significant vibration by their very slowness. They are faster than prefiltering because of the time delay inherent in a prefiltered move. Long moves, where the robot travels at a maximum speed for a large percentage of the move distance, are the ones most conducive to filtering techniques.

Second, we can go a long way towards eliminating residual vibration by using a gentle acceleration or an acceleration that takes the same amount of time as a full cycle of vibration at the natural frequency. It may not eliminate all of the vibration and it may not be the fastest way to get there, but it does a lot of the work for you and it is a very traditional method of dealing with systems that vibrate.

Third, prefiltering can do a lot for eliminating residual vibration. In cases where you know the vibrational frequencies and they do not change during the course of

the move, prefiltering works quite well. Prefiltering doesn't take care of all of the problem: higher modes of vibration may be excited. These can be dealt with by more complicated filters or by mixing reasonable accelerations with prefilters.

Fourth, even a complex move with changing vibrational frequencies can be improved. Careful prefiltering and reasonable velocities can reduce the amount of residual vibration in just about any motion to almost zero. Unfortunately, there are tradeoffs involved. Cutting the residual vibration to almost nothing slows the move down, while getting there at a fast speed increases the residual vibration. The optimal implementation depends strongly on whether it is important to get there fast or to get there with little vibration.

Fifth, cartesian motion filtering is still an open question. Although prefiltering definitely can reduce the amount of vibration felt during a cartesian move, the best techniques to use are still uncertain.

Summary and Future Work

Chapter 6

6.1 Summary

A three degree of freedom flexible robot has been designed, built, and controlled. The robot exhibits frequencies of vibration as low as 2.0 hertz and the operating parameters are easy to modify. Preliminary experiments in the reduction of vibration left after a move has been completed have met with success in both simple and complicated motions.

Chapter 2 demonstrated a simple correlation between the endpoint deflection of the robot under gravity loading and the lowest natural frequency of vibration. To minimize the deflection and keep the frequency as low as possible it is best to maximize the weight of the payload with respect to the weight of the elbow joint and the links. Material choice is important for a flexible robot because strong materials are often stiffer than weak materials. Links made out of strong materials must be thinner to get the same vibrational frequencies, leading to higher stresses.

The robot has three rotary actuators and two links; two of the actuators are located in the base and the third sits between the links as an elbow joint. The two actuators at the base consist of a DC servo motor connected to a 10:1 timing belt reduction. The elbow joint uses a DC torque motor connected to a 5:1 planetary gear set. Position information is read from optical encoders connected to the motor

shafts. All three joints use stacks of spring washers between the gear reduction and the output to allow the operator to modify the flexibility of the joint. The links are designed to be easily interchangeable. The robot carries a 3 pound payload and has a peak acceleration of over 240 ft/sec^2 at the tip.

The robot is controlled by three 68000 based processor boards and a number of interface boards on a VMEbus backplane. A Sun 3/180 workstation running the Condor system provides the development environment and disk storage. A convenient interface allows the operator to selectively enable and disable the amplifiers and fail-safe brakes.

A software control package has been implemented that controls the robot in either joint space or endpoint cartesian space. A PD servo loop with a full order observer to reconstruct joint velocities servos all three axes. The robot takes two primary types of movement commands: velocity commands and acceleration commands. Velocity commands are a series of equally spaced setpoints that assume infinite acceleration. Acceleration commands are a series of setpoints which accelerate the robot up to speed, cruise at the maximum velocity, and then decelerate down to zero. Teleoperation is implemented with a hand held joystick box that allows endpoint velocity control of the robot in cartesian space.

Experiments have been run to determine how the arm behaves and to try to eliminate residual vibration using impulse prefilters. The arm has four low frequencies of vibration that have been mapped and vary with the position of the arm. The first series of experiments deals with controlling the residual vibration in a long move where the vibrational frequencies do not change over the course of the move. Acceleration time is important in determining the amount of residual vibration excited by the move. Timing the acceleration to mimic the effects of a simple two impulse prefilter gave a low amount of residual vibration.

Experiments using prefilters and combinations of accelerations and prefilters

showed that the residual vibration for a simple move could be virtually eliminated without sacrificing overall move time. The best methods used several prefilters at the natural vibrational frequencies or a combination of a prefilter and an acceleration move to eliminate residual vibration and still move the arm quickly.

More complicated motions of the arm are more difficult to control. A series of experiments were run where the vibrational frequencies at the starting position of the arm were different from those at the finishing position. Using filtered accelerations or several prefilters cut the residual vibration of the arm almost to zero at low speeds, but not at higher speeds. The overall time it took the arm to make the move and stop vibrating was relatively constant for most of the types of vibration control schemes; some cut down the residual vibration but ran slower and some got there quickly but had to wait for the vibrations to damp out.

Finally, a few experiments in cartesian motion were performed. The results are inconclusive. A better method of finding the position of the endpoint of the robot needs to be implemented before we can perform comprehensive experiments in cartesian motion.

6.2 Future Work

Now that the robot has been built, there are almost endless possibilities for experiments that could be done. There are also a number of simple tasks that would improve the robot and make data collection easier. Suggestions and ideas are listed in order from the mundane to the speculative.

To start, the arm could use a few new sets of links made out of different materials (such as fishing poles) and of different shapes and sizes. The microswitch positioning scheme mentioned in Chapter 4 should be fixed so that the arm would be able to reliably locate itself in absolute coordinates before taking data. There are still occasional problems with electrical noise. These could be eliminated by moving

the 28 volt power supply to the base of the robot and rewiring the optical encoder circuits.

For future experiments it would be useful to have better techniques of identifying how the arm is behaving rather than the current scheme of reading the positions of the motors. Accelerometers and strain gauges mounted on the links could be used for both raw data gathering and in adaptive control systems. Some form of optical reading of the tip position by cameras or by CCD arrays mounted along the robot could give precise information about the position of the robot.

A simple manipulator or gripper mounted at the end of the robot would be useful for experiments on changing the frequency of vibration by adding a large payload. It could also be used for moving around a flexible payload that would add additional modes of vibration to the system, or for interacting with walls and other immobile objects.

The vibrational behavior of the arm is complex; it would be useful to have a complete modal analysis of the arm including mode shapes, damping ratios, and transfer functions between the output of a motor and the resulting motion. Such a mapping could be actively used by the robot for controlling the frequency settings of prefilters.

A complete dynamic model and simulation of the robot could be compared to the modal analysis. Such a simulation would allow off-line testing of new control schemes and quick experimentation with different link geometries and payloads. This information would also be useful in the important task of designing a better controller for the robot. The current controller does not take into account the system dynamics. A more complete controller should lead to improved performance and better trajectory tracking.

The open loop vibration control schemes can be improved. A generalized approach that utilizes the basic concept of shaping the input so that the robot gets

up to speed without vibration might be implemented to track the position of the arm and select accelerations accordingly. Work also needs to be done in controlling centripetally induced vibration.

Closed loop schemes incorporating feedback from the robot and prefilters could reduce the vibration felt during the move and actively reject vibration caused by outside sources. Switching the control schemes between long moves and servoing in place might turn out to be the best way to handle both problems. Adaptive control has great possibilities.

One concern with the robot is of the vibration felt during a complicated move. Intelligent planning schemes could be used to calculate the volume in which the vibration of the robot will be confined and use that to select trajectories for the robot to follow. Better control systems might alter the input commands to keep the stress level in the links to a minimum during the move.

Finally, the flexbot has interesting possibilities in force control. Most robots are stiff and with conventional control schemes they often limit cycle when trying to exert a constant force on the environment. But the flexbot is not stiff and hence it might be easier for it to maintain a force without losing contact.

Bibliography

[1] **Alberts, T.E., Hastings, G.G., Book, W.J., and Dickerson, S.L.,**
"Experiments in Optimal Control of a Flexible Arm with Passive Damping", *Fifth VPISSU/AIAA Symposium on the Dynamics and Control of Large Structures*, 1985.

[2] **Bayo, E.,**
"Computed Torque for the Position Control of Open-Chain Flexible Robots", *IEEE International Conference on Robotics and Automation*, April 25–29, 1988.

[3] **Bayo, E.,**
"A Finite-Element Approach to Control the End-Point Motion of a Single-Link Flexible Robot", *Journal of Robotic Systems*, Vol. 4, No. 1, 1987.

[4] **Bayo, E. and Moulin, H.,**
"An Efficient Computation of the Inverse Dynamics of Flexible Manipulators in the Time Domain", *IEEE International Conference on Robotics and Automation*, 1989.

[5] **Blevins, R.D.,**
Formulas for Natural Frequency and Mode Shape, Van Nostrand Reinhold Company, Appendix A, 1979.

[6] **Cannon, Jr., R.H. and Schmitz, E.,**
"Initial Experiments on the End-Point Control of a Flexible One-Link Robot", *The International Journal of Robotics Research*, Vol. 3, No. 3, Fall 1984.

[7] **Chalhoub, N.G. and Ulsoy, A.G.,**
"Control of a Flexible Robot Arm: Experimental and Theoretical Results", *Journal of Dynamic Systems, Measurement, and Control*, Vol. 109, December 1987.

[8] **Chrétien, J-P.,**
"SECAFLEX: An Experimental Set-up for the Study of Active Control of Flexible Structures", *CERT/DERA*, 1989.

[9] *Control of Flexible Structures (COFS) Technology Program*, NASA Langley Research Center, Hampton, VA, Februrary 1986.

[10] **Daniel, R.W., Irving, M.A., Fraser, A.R. and Lambert, M.,**
"The Control of Compliant Manipulator Arms", *4th International Symposium on Robotics Research*, MIT Press, 1987.

[11] **Feliu, V., Rattan, K.S., and Brown, Jr., H.B.,**
"Adaptive Control of a Single-Link Flexible Manipulator in the Presence of Joint Friction and Load Changes", *IEEE International Conference on Robotics and Automation*, May 14–19, 1989.

[12] **Hollars, M.G. and Cannon, Jr., R.H.,**
"Initial Experiments on the End-Point Control of a Two-Link Manipulator with Flexible Tendons", *ASME Winter Annual Meeting*, November 19, 1985.

[13] **Koivo, A.J. and Lee, K.S.,**
"Self-Tuning Control of Planar Two-Link Manipulator with Non-Rigid Arm", *IEEE International Conference on Robotics and Automation*, May 14–19, 1989.

[14] **Kotnik, P.T., Yurkovich, S., and Özgüner, Ü.,**
"Acceleration Feedback for Control of a Flexible Manipulator Arm", *Journal of Robotic Systems*, Vol. 5, No. 3, 1988.

[15] **Meckl, P.H.,**
"Control of Vibration in Mechanical Systems Using Shaped Reference Inputs", *Ph.D. Thesis, also AI Technical Report #1018* , MIT Artificial Intelligence Laboratory, February, 1988.

[16] **Narasimhan, S., Siegel, D.M., and Hollerbach, J.M.,**
"A Standard Architecture for Controlling Robots", *MIT Artificial Intelligence Laboratory Memo AIM 977*, July, 1988.

[17] **Narasimhan, S., Siegel, D.M.,**
"The Condor Programmer's Manual — Version II", *MIT Artificial Intelligence Laboratory Working Paper 297*, July, 1987.

[18] **Narasimhan, S., Siegel, D.M., and Hollerbach, J.M.,**
"Condor: A Revised Architecture for Controlling the Utah-MIT Hand", *IEEE International Conference on Robotics and Automation*, April 25–29, 1988.

[19] **Oakley, C.M. and Cannon, Jr., R.H.,**
"Initial experiments on the control of a two-link manipulator with a very flexible forearm", *IEEE International Conference on Robotics and Automation*, April 25–29, 1988.

[20] **Petterson, B.J. and Robinett, R.D.,**
"Model Based Damping of Coupled Horizontal and Vertical Oscillations in a Flexible Rod", *AIAA Guidance, Navigation, and Controls Conference, Boston, MA*, August 14–16, 1989.

[21] **Pfeiffer, F. and Gebler, B.,**
"A Multistage-Approach to the Dynamics and Control of Elastic Robots", *IEEE International Conference on Robotics and Automation*, April 25–29, 1988.

[22] **Quinn, R.D. and Meirovitch, L.,**
"Maneuver and Vibration Control of SCOLE", *Journal of Guidance and Control*, Vol. 11, No. 6, 1988.

[23] **Schmitz, E.,**
"Modeling and Control of a Planar Manipulator with an Elastic Forearm", *IEEE International Conference on Robotics and Automation*, May 14–19, 1989.

[24] **Singer, N.C. and Seering, W.P.,**
"Preshaping Command Inputs to Reduce System Vibration", *Submitted to the ASME Journal of Dynamic Systems, Measurement and Control*, March, 1988.

[25] **Singer, N.C.,**
"Residual Vibration Reduction in Computer Controlled Machines", *Ph.D. Thesis, also AI Technical Report #1030* , MIT Artificial Intelligence Laboratory, February, 1989.

[26] **Thomson, W.T.,**
Theory of Vibration with Applications, pp 174–175, Prentice-Hall, Inc., 1981.

[27] **Tilley, S.W., Cannon, Jr., R.H., and Kraft, R.,**
"End Point Force Control of a Very Flexible Manipulator with a Fast End Effector", *ASME Winter Annual Meeting*, December, 1986.

[28] **Tzes, A.P., Englehart, M.J., and Yurkovich, S.,**
"Input Preshaping with Frequency Domain Information for Flexible-Link Manipulator Control", *AIAA Guidance, Navigation and Control Conference*, August, 1989.

[29] **Wie, B.,**
"Active Vibration Control Synthesis for the Control of Flexible Structures Mast Flight System", *Journal of Guidance and Control*, Vol. 11, No. 3, May–June 1988.

[30] **Yuh, J., Young, T., and Baek, Y.S.,**
"Modeling of a Flexible Link Having a Prismatic Joint in Robot Mechanism — Experimental Verification", *IEEE International Conference on Robotics and Automation*, May 14–19, 1989.

[31] **Yurkovich, S. and Pacheco, F.E.,**
"On Controller Tuning for a Flexible-Link Manipulator with Varying Payload", *Journal of Robotic Systems*, Vol. 6, No. 3, 1989.